Desenvolvimento da Interface Gráfica para o SysObj (Ferramenta de Co-projecto)

Nelson Pereira, I.C Teixeira, J.P. Teixeira, O. P. Dias

IST / INESC-ID Lisboa Tech. Report 3/2002
ProSys Laboratory

December, 2002

Índice

1. Introdução 1
2. SysObj *v4.0* - Manual de Instalação 1
Notação 1
2.1. Introdução 2
2.1.1. Configuração Mínima de Hardware/Software 2
2.1.2. Conteúdo do CD de Instalação 2
2.2. Procedimentos de Instalação 3
2.2.1. Antes de Instalar 3
2.2.2. Procedimentos 4
3. SysObj *v4.0* - Manual do Utilizador 9
Resumo 9
Notação 9
3.1. Introdução 11
3.1.1 Conceitos Utilizados 11
3.1.2 Sequências Lógicas de Utilização 13
3.2. Ambiente de Trabalho 13
3.2.1. Barras de Ferramentas 14
3.2.1.1. Barra de Ferramentas do Projecto 14
3.2.1.2. Barra de Ferramentas da Implementação Gráfica 18
3.2.2. Lista de Objectos Existentes 22
3.2.3. Área de Implementação 24
3.2.4. Visualização das Áreas de Implementação 26
3.2.5. Menu de Atalhos 27
3.3. Métodos de Trabalho 29
3.3.1. Desenho de Objectos 29
3.3.2. Selecção de Objectos 30
3.3.3. Mover / Redimensionar Objectos 30
3.3.4. Eliminar Objectos 31
Bibliografia 32
4. SysObj *v4.0* – Manual Técnico 33
Resumo 33
Notação 33
Glossário 36
4.1. Introdução 38
4.1.1. Enquadramento Teórico 39
4.1.2 Ferramentas Existentes 43
4.1.3 Interface Gráfica 45
4.2. Estruturas de Dados 45
4.2.1. Tabelas 45
4.3. Descrição de Use-cases 47
4.3.1 Interligação entre os DFDs e as Matrizes do `SysObj` 47
4.3.1.1. *Use-case* de Sincronização 47
4.3.1.2. Processo “Sincronizar Matrizes com DFD” 49
4.3.2. Desenho de Objectos na Interface Gráfica 51
4.3.2.1. *Use-case* de Desenho de um Actor, Processo, Arquivo ou Texto 52
4.3.2.2. *Use-case* do Desenho de um Atributo 53

4.4. Demonstração de Resultados / Exemplo de Utilização: Sistema FILL.....57
4.4.1. Análise do Sistema.....57
4.4.2. Exemplo de Utilização.....63
4.4.3 Demonstração de Resultados.....78
4.5. Conclusões.....90
4.5.1. Trabalho Realizado.....90
4.5.2. Perspectivas de Trabalho Futuro.....91
Bibliografia.....92

1. Introdução

O objectivo deste documento é apresentar, em sequência, o Manual de Instalação, o Manual de Utilizador e o Manual Técnico da versão actual (4.0) da ferramenta computacional `SysObj`, uma ferramenta para suporte ao projecto e teste de sistemas *hardware / software* descritos em alto nível de abstracção através das técnicas de modelação por objectos (OOM, *Object-Oriented Modeling*).

2. SysObj *v4.0* - Manual de Instalação

Notação

Ao longo deste manual foram utilizados diferentes estilos gráficos, de forma a facilitar a interpretação dos vários conceitos. A notação utilizada foi a seguinte :

Expressões em inglês estão apresentadas em *itálico*

Conceitos que se pretendem destacar são apresentados em bold

Nomes de ferramentas estão representados em `Courier New`

O restante texto está apresentado em Times New Roman

Lista de Figuras

2.2.1 – Advertência à instalação do `SysObj` 6
2.2.2 – Janela de introdução á instalação 6
2.2.3 - Confirmação do cancelamento da instalação 7
2.2.4 – Directoria de instalação 7
2.2.5 – Escolha da directoria de instalação 8
2.2.6 – Escolha do tipo de instalação 9
2.2.7 – Escolha dos componentes a instalar 9
2.2.8 – Escolha da localização do atalho para `SysObj` 10
2.2.9 – Resumo e confirmação das opções escolhidas 10

2.1. Introdução

Neste capítulo estão descritos os aspectos relativos ás configurações necessárias á instalação do SysObj. É ainda apresentada uma lista do conteúdo do CD do SysObj onde se pode encontrar a localização do programa de instalação cujos procedimentos irão ser descritos no capítulo seguinte.

2.1.1. Configuração Mínima de Hardware/Software

O SysObj foi desenvolvido para o sistema operativo *Windows* podendo ser exucutado em qualquer versão deste sistema apartir da versão *Windows 95*. Assim sendo o SysObj pode ser instalado nas seguintes plataformas:

Windows 95/98
Windows NT
Windows 2000
Windows Milenium
Windows XP

Este é o único requisito mínimo de software necessário á instalação do SysObj.
A nível de *Hardware* sugerem-se duas configurações, a mínima e a aconselhada.
Assim,

Configuração mínima:

Processador Pentium® 100Mhz ou equivalente
32MB de memória RAM
50MB de espaço livre de Disco Rígido

Configuração aconselhada:

Processador PentiumII® 450Mhz ou equivalente
64MB de memória RAM
100MB de espaço livre de Disco Rígido

2.1.2. Conteúdo do CD de Instalação

O CD do SysObj é constituído pelos seguintes componentes :
SysObj v4.0:

Programa de instalação
Código fonte
Manuais :
Manual do Utilizador – Interface Gráfica
Manual de Instalação – SysObj v4.0
Manual Técnico – Interface Gráfica
Bases de dados de exemplo:
Use case Fill_1

Use case Fill_2
Use case da Interface Gráfica do `SysObj`

`SysObj v1.0`:
Programa de instalação
Código fonte
Manuais:
Introdução Manual de Projectos
Listagem
Manual Técnico – Parte 1
Manual Técnico – Parte 2
Manual Técnico – Anexo 2
Relatório

2.2. Procedimentos de Instalação

Neste capítulo estão descritos os aspectos relativos á instalação do `SysObj` assim como alguns cuidados a ter nomeadamente no que respeita á coexistência do `SysObj` com outros programas que utilizem a mesma plataforma de bases de dados, isto é, o BDE (*Borland Database Engine*). São ainda descritos todos os passos necessários para a correcta instalação do `SysObj`, utilizando para isso as imagens das janelas de diálogo de todo o procedimento de instalação.

2.2.1. Antes de Instalar

Um problema que se coloca na instalção do `SysObj` e em todas as instalações de programas com recurso a bases de dados que utilizam a plataforma BDE (Borland Database Engine) prende-se com o facto de uma instalação poder vir a intreferir com o normal funcionamento das outras aplicações. Isto acontece porque a instalação total do BDE sobrepõe todas as configurações existentes de outras bases de dados.

Assim sendo, se o computador onde vai ser instalado o `SysObj` já tiver aplicações que utilizem esta plataforma terá que ser escolhido tipo de instalação (Figura 2.6) "*Compact*" ou o tipo "*Custom*". No caso de ser escolhido o tipo "Custom" na janela de diálogo seguinte não deve ser seleccionada a opção "*BDE Configuration Files*".

Para advertir o utilizador é apresendada durante a instalação uma janela (Figura 2.2.1) com as informações necessárias para efecturar uma correcta instalação.

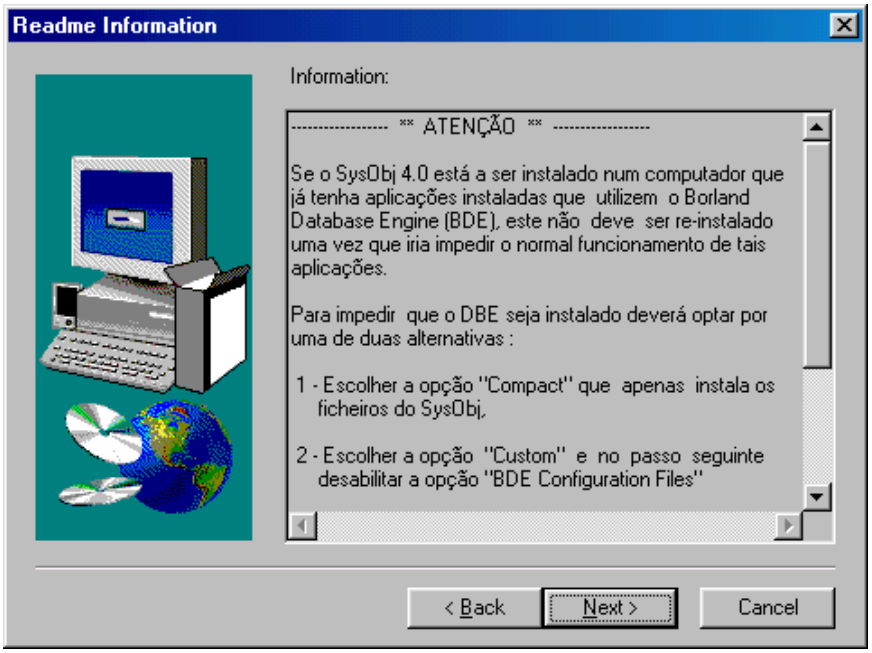
Readme Information
Information:
---------- ** ATENÇÃO ** ----------
Se o SysObj 4.0 está a ser instalado num computador que já tenha aplicações instaladas que utilizem o Borland Database Engine (BDE), este não deve ser re-instalado uma vez que iria impedir o normal funcionamento de tais aplicações.
Para impedir que o DBE seja instalado deverá optar por uma de duas alternativas :
1 - Escolher a opção "Compact" que apenas instala os ficheiros do SysObj,
2 - Escolher a opção "Custom" e no passo seguinte desabilitar a opção "BDE Configuration Files"
< Back
Next >
Cancel

Figura 2.2.1 – Advertência à instalação do `SysObj`

2.2.2. Procedimentos

Ao executarmos o programa de instalação que se encontra no CD de instalação conforme descrito no ponto 2.1.2, é apresentada a primeira janela de diálogo que faz uma introdução ao programa de instalação (Figura 2.2.2) e adverte para o facto de ser recomendado que todos os progamas do *Windows* sejam encerrados antes de continuar o procedimento de instalação. Para prosseguir a instalação pressione o botão "*Next*".

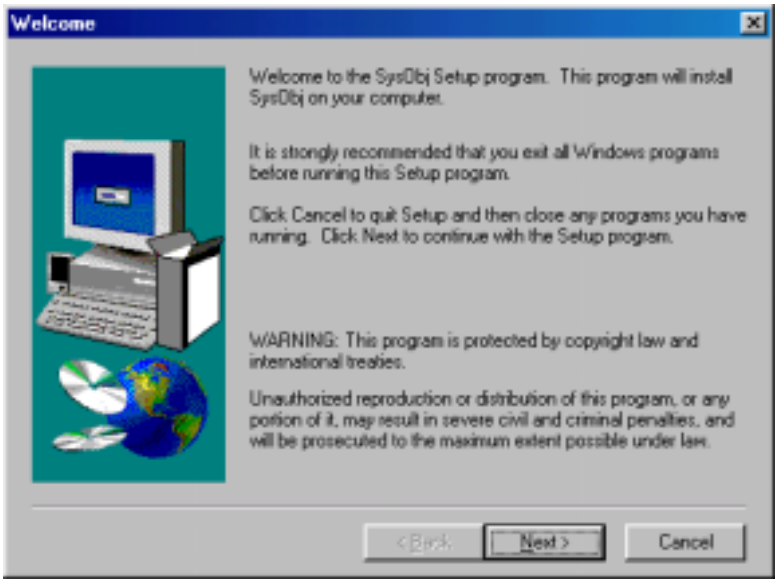
Welcome
Welcome to the SysObj Setup program. This program will install SysObj on your computer.
It is strongly recommended that you exit all Windows programs before running this Setup program.
Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.
WARNING: This program is protected by copyright law and international treaties.
Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.
< Back
Next >
Cancel

Figura 2.2.2 – Janela de introdução á instalação

Posteriormente é apresentada uma janela com algumas advertências sobre o procedimento de instalação (Figura 2.2.1). Para melhor compreender esta janela consultar o ponto 2.2.1.

Em qualquer fase é possível cancelar a instalação, bastando para isso pressionar o botão "*Cancel*" o que dá origem a uma janela de confirmação (Figura 2.2.3) onde é necessário confirmar a intenção de abandonar a instalação.

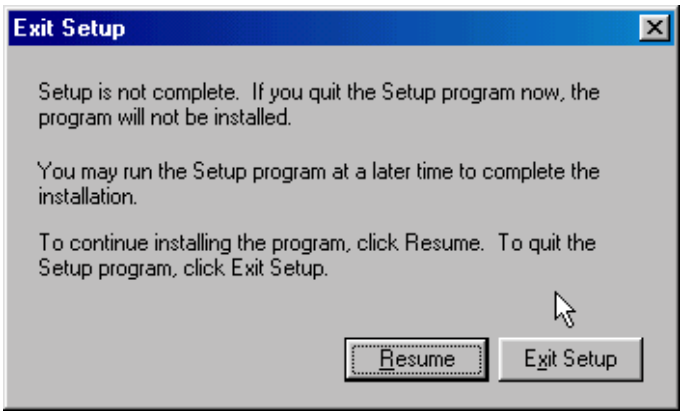
Exit Setup
Setup is not complete. If you quit the Setup program now, the program will not be installed.
You may run the Setup program at a later time to complete the installation.
To continue installing the program, click Resume. To quit the Setup program, click Exit Setup.
Resume
Exit Setup

Figura 2.2.3 – Confirmação do cancelamento da instalação

Para avançar é sempre necessário pressionar o botão "*Next*". É ainda possível voltar passos atrás pressionando o botão "Back". Quando for feito "Next" na janela que antecede a instalação (Figura 2.2.9) já não é possível voltar atrás com o procedimento. Apesar disso é ainda possível cancelar a instalação.

Na janela seguinte (Figura 2.2.4) pode ser escolhida a directoria onde irá ser instalado o `SysObj`. É sugerida um directoria por omissão, porém esta pode ser alterada, bastando para isso fazer *click* no botão "*Browse...*"

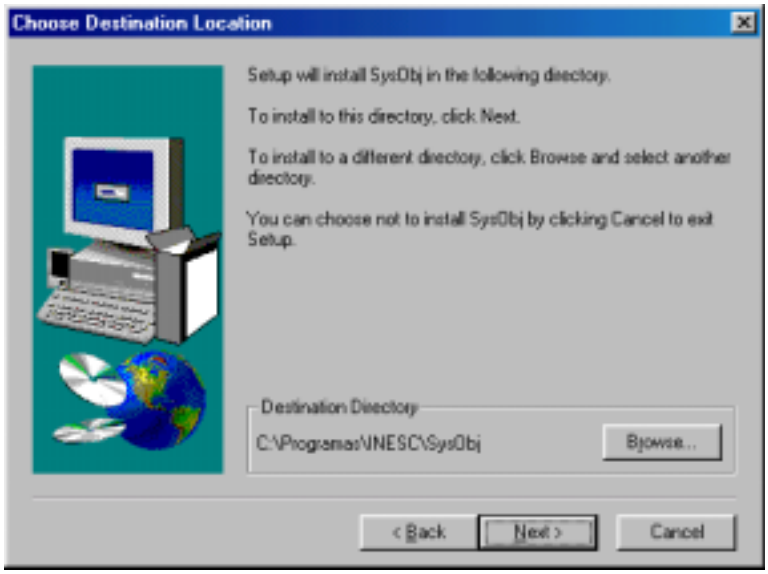
Choose Destination Location
Setup will install SysObj in the following directory.
To install to this directory, click Next.
To install to a different directory, click Browse and select another directory.
You can choose not to install SysObj by clicking Cancel to exit Setup.
Destination Directory
C:\Programas\UNESC\SysObj
Browse...
< Back
Next >
Cancel

Figura 2.2.4 – Directoria de instalação

Se foi pressionado o botão "*Browse...*" é apresentada uma nova janela (Figura 2.2.5) onde pode ser introduzida uma directoria na caixa de texto ou pode ser escolhida utilizando a lista de directorias apresentada.

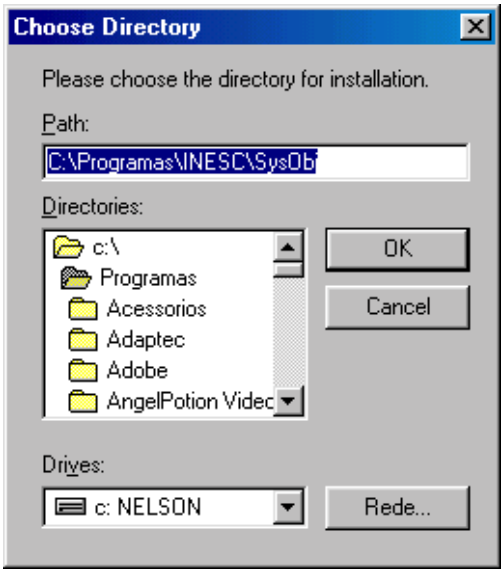

Figura 2.2.5 – Escolha da directoria de instalação

No fim de ser escolhida uma directoria, ou de aceitar a directoria sugerida, e de pressionar o botão "*Next*" surge a caixa de diálogo onde é seleccionado o tipo de instalação desejado (Figura 2.2.6).

Antes de proceder com a instalação aconselha-se a leitura do ponto 2.2.1 deste manual.

Assim nesta caixa de diálogo são apresentados os seguintes tipos de instalação :

- "*Typical*" , nesta opção são instalados todos os componentes necessários ao `SysObj`, *drivers* gráficos e ficheiros de configuração do BDE (*Borland Database Engine*), e ainda as bases de dados de exemplo.

- "*Compact*" , apenas são instalados os ficheiros necessários ao `SysObj`, não são instaladas as bases de dados de exemplo.

- "*Custom*", nesta opção é apresentada uma nova janela (Figura 2.2.7) onde é possível escolher os componentes que desejamos instalar.

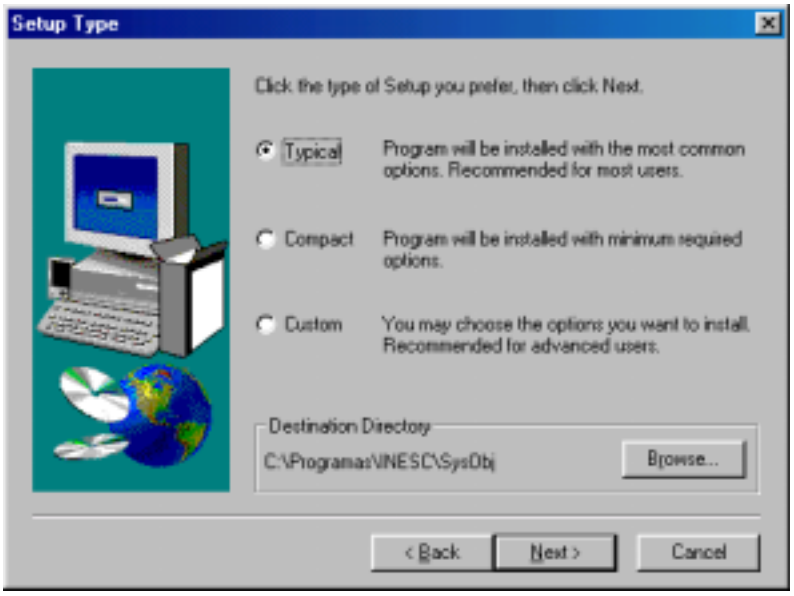
Setup Type
Click the type of Setup you prefer, then click Next.
Typical
Program will be installed with the most common options. Recommended for most users.
Compact
Program will be installed with minimum required options.
Custom
You may choose the options you want to install. Recommended for advanced users.
Destination Directory
C:\Programas\INESC\SysObj
Browse...
< Back
Next >
Cancel

Figura 2.2.6 – Escolha do tipo de instalação

Na janela de escolha de componentes a instalar (Figura 2.2.7) são apresentados todos o componentes que compõem a instalação do `SysObj` Para um componente ser escolhido tem que estar marcado com uma "*check*" na caixa criada para o efeito. Por defeito todas as opções estão selecionadas, o que corresponde á opcção "*Typical*" da janela anterior (Figura 2.2.6).

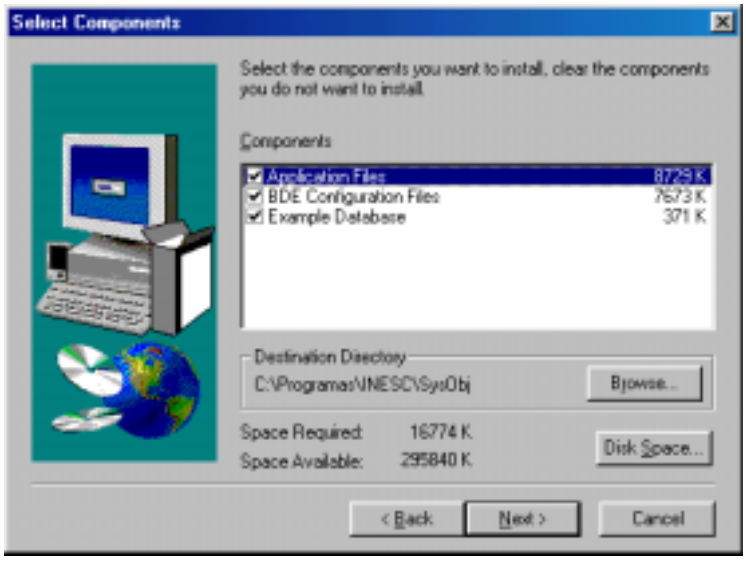

Figura 2.2.7 – Escolha dos componentes a instalar

No passo seguinte (Figura 2.2.8) é possível escolher a pasta do menu iniciar (*Start Menu*) onde vai ser colocada a pasta do `SysObj`. É ainda possível alterar o nome da pasta a ser criada.

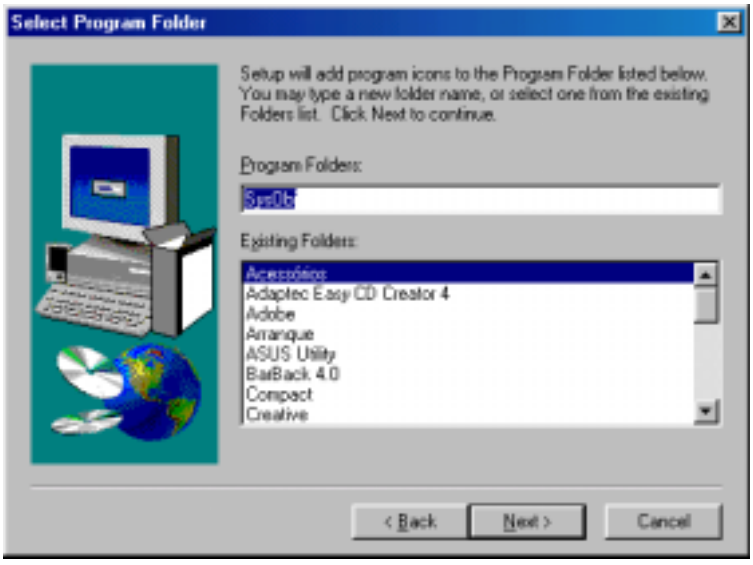

Figura 2.2.8 – Escolha da localização do atalho para `SysObj`

Finalmente, é apresentada a janela com um resumo das opções selecionadas durante o procedimento de instalação (Figura 2.2.9), ao fazer "*Next*" a instalação é iniciada e já não é possível alterar opções de intalação. Se está a instalar uma versão de disquetes ser-lhe-á pedido que introduza as restantes disquetes do programa.

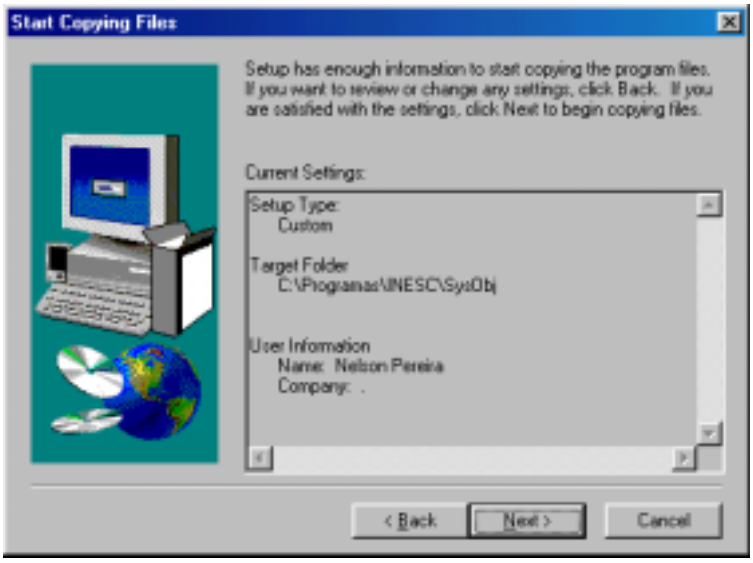
Start Copying Files
Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Next to begin copying files.
Current Settings:
Setup Type:
Custom
Target Folder
C:\Programas\INESC\SysObj
User Information
Name: Nelson Pereira
Company: .
< Back
Next >
Cancel

Figura 2.2.9 – Resumo e confirmação das opções escolhidas

3. SysObj *v4.0* - Manual do Utilizador

Resumo

A versão 4.0 do `SysObj` foi desenvolvida com o objectivo de criar uma interface gráfica para o desenho de Diagramas de Fluxos de Dados (DFDs), dando assim ao projectista uma total independência das ferramentas CAD para o desenvolvimento dos mesmos DFDs. Assim o projectista pode realizar todo o trabalho de modelação do projecto, a partir do desenho dos DFDs, utilizando apenas a mesma ferramenta, o SysObj 4.0.

Nota: Quando for referenciado – `SysObj` – entenda-se como sendo a versão `4.0`

Notação

Ao longo deste manual foram utilizados diferentes estilos gráficos, de forma a facilitar a interpretação dos vários conceitos. A notação utilizada foi a seguinte :

Expressões em inglês estão apresentadas em *itálico*
Conceitos que se pretendem destacar são apresentados em bold
Nomes de ferramentas estão representados em `Courier New`
O restante texto está apresentado em Times New Roman

Lista de Figuras

3.1.1 - Menu do `SysObj` 13

3.2.1. - Áreas distintas da interface 14
3.2.2 - Barra de ferramentas do projecto 14
3.2.3 - Sair 15
3.2.4 - Guardar alterações 15
3.2.5 - Nova área de implementação 15
3.2.6 - Gravar 15
3.2.7 - Copiar para a área de transferência 15
3.2.8 - Anular 16
3.2.9 - Repor última gravação 16
3.2.10 - Visualizar áreas de implementação 16

3.2.11 - Apagar área de implementação 16
3.2.12 - Apagar objectos existentes 17
3.2.13 - Teste de coerência 17
3.2.14 - Sincronizar bases de dados 17
3.2.15 - Alternar entre módulos do projecto 17
3.2.16 - Gravar alterações 18
3.2.17 - Barra de ferramentas da implementação gráfica 18
3.2.18 - Cursor 18
3.2.19 - Actor 18
3.2.20 - Processo 18
3.2.21 - Arquivo 18
3.2.22 - Atributo 19
3.2.23 - Texto 19
3.2.24 - Espessura da linha 19
3.2.25 - Diferentes espessuras de linha 19
3.2.26 - Estilo da linha 19
3.2.27 - Diferentes estilos de linha 19
3.2.28 - Cor da linha 20
3.2.29 - Diferentes cores de linha 20
3.2.30 - Cor de fundo 20
3.2.31 - Diferentes cores de fundo 21
3.2.32 - Estilo e cor do texto 23
3.2.33 - Diferentes estilos e cores do texto 22
3.2.34 - Apagar objecto seleccionado 22
3.2.35 - Lista de objectos existentes 23
3.2.36 - Actores 23
3.2.37 - Atributos 23
3.2.38 - Arquivos 23
3.2.39 - Processos 23
3.2.40 - Agrupamento de objectos seleccionado 24
3.2.41 - Área de implementação 24
3.2.42 - Visualização das áreas de implementação 26
3.2.43 - Lista de objectos ligados 27

3.2.44 - Lista de atributos.....27
3.2.45 - Menu de atalhos 28
3.2.46 - Propriedades de um objecto 28
3.2.47 - Propriedades de um atributo.....28
3.2.48 - Nome da área de desenho/implementação 29

3.3.1 - Apagar todas as implementações do objecto 31
3.3.2 - Apagar atributos do objecto 32

3.1. Introdução

Neste capítulo é feita a explicação dos termos utilizados na interface gráfica do `SysObj`, isto para que o utilizador não interprete de forma incorrecta alguns dos conceitos utilizados, uma vez que o seu significado por vezes é discutível.

É ainda feita uma abordagem sobre o funcionamento da interface e qual a sequência lógica de utilização para que a interface gráfica corresponda às expectativas do utilizador.

3.1.1 Conceitos Utilizados

Na realização do projecto utilizando a interface gráfica do `SysObj` são utilizados conceitos cujo significado se descreve em seguida :

Actor (*Actor*) – Entidades que interagem com os processos, activam processos, recolhem informações dos processos, etc. Em sistemas industriais os actores podem ser vistos como sendo os operadores das máquinas.

Atributo (*Atribute*) – Podem ser atributos de entrada ou saída dos processos. Os atributos são as ligações dos processos a actores ou a bancos de dados. Estes transportam, ou dados no caso de serem atributos de dados (*Data Atributes*), ou acções, ordens de execução, no caso de serem atributos de controlo (*Control Atributes*). Um objecto do tipo atributo não tem agrupamento em classes uma vez que não fazia sentido ter várias implementações de um mesmo fluxo de dados ou de controlo.

Classe (*Class*) – Agrupamento de Objectos, uma classe agrupa uma ou mais implementações de objectos. Por exemplo se eu tiver uma classe de Actores chamada “Sensores” eu posso ter várias implementações do Actor “Sensores” que podem ser, “Sensor Temperatura”, “Sensor de Humidade”, etc. Estes dois actores são dois

objectos que representam implementações diferentes da classe “Sensores” e como tal têm atributos diferentes e entram (ou não) em processos diferentes.

Dados (*Data*) – Bancos de dados, estes objectos guardam informação necessária a actores ou a processos. Estas informações podem ser obtidas por actores e processos e actualizadas através dos processos, sempre através de atributos.

Use Case – Representa a funcionalidade de uma dada tarefa ou procedimento ou num cenário mais completo, uma das funcionalidades de um sistema complexo. Por exemplo se existir um projecto para medir temperaturas e tomar acções consoante as medidas pode-se considerar a funcionalidade de medição um *use case* e a funcionalidade de tomar acções outro *use case*. Os *use cases* são usualmente as primeiras definições de um projecto quando ainda não existe uma ideia bem clara sobre como cada funcionalidade deve ser executada. Isto deve-se ao elevado poder de abstracção dos *use cases*.

DFD – Diagrama de Fluxos de Dados (*Data Flow Diagram*), servem para descrever funcionalidades. Tomando o exemplo anterior pode ser definido um DFD para o *use case* de medir temperaturas. Os DFDs têm vários níveis de abstracção, isto é, começando com o DFD de nível 0, nível mais abstracto, que descreve uma funcionalidade pode-se ir detalhando cada funcionalidade nova que vai surgindo num outro DFD de nível mais baixo. Estas funcionalidades novas vão surgindo quando se está a detalhar uma outra funcionalidade de nível de abstracção superior. Assim pode-se equiparar um *use case* a um DFD de nível 0 porque têm níveis de abstracção equivalentes.

DFD coerente – Para um DFD estar coerente têm que estar cumpridas as seguintes condições:

Cada objecto tem que ter pelo menos um atributo, seja um atributo de entrada ou de saída.
Cada classe tem que ter atributos de entrada e de saída (Tendo em conta todos os objectos dessa classe).

Área de Implementação (*Implementation area*) – Área onde são implementados/desenhados os objectos. Estas áreas podem ser utilizadas para criar *use cases* separados, com diferentes níveis de detalhe. Por exemplo numa primeira área eu utilizo um processo da classe “xpto” e numa área distinta vou então descrever o processo “xpto” com mais detalhe, ou seja vou ter uma nova implementação da classe “xpto” que tem mais atributos, mais consultas a dados, etc. Esta separação é apenas lógica e só serve para ajudar o projectista na esquematização do projecto uma vez que o DFD é global.

Objecto (*Object*) – Define a implementação de uma figura no diagrama de fluxo de dados, este objecto pertence a uma dada classe, podendo existir mais objectos que sejam outras implementações da mesma classe. Um objecto pode ser, um Actor (*Actor*), um Processo (*Process*) ou um Atributo (*Atribute*), se bem que neste último caso (atributos) não existe a noção de classe.

Processo (*Process*) – Processam informação através de dados fornecidos por actores ou por bancos de dados. Os atributos de controlo a partir de actores ou de outros processos activam os processos.

De notar que os botões nos diálogos do sistema aparecem em Português, isto deve-se ao facto de estas imagens terem sido recolhidas numa versão do Windows em Português, estas caixas de diálogo são geradas pelo Windows com a língua corrente. O mesmo se aplica aos diálogos de selecção de cores e de formatação de texto.

3.1.2 Sequências Lógicas de Utilização

Figura 3.1.1 – Menu do `SysObj`

Para entrar fazer (Project|DFD Graphic Interface). Ao entrar nesta interface todos as outras opções do `SysObj` ficam desactivadas. É necessário sair da interface gráfica para activar as outras opções do `SysObj`.

Se o projecto actual já tiver diagramas de fluxos de dados desenhados eles serão abertos nas áreas de implementação em que foram criados para que o projectista possa fazer alterações, caso contrário é criada uma nova área de implementação vazia para dar início ao desenho da funcionalidade em questão.

Para que a informação contida num DFD seja registada nas matrizes que descrevem o sistema é necessário executar o comando *sincronize*, que está destrito mais adiante neste manual. Quando este comando é executado o `SysObj` testa a coerência do projecto e a nova informação só é registada se estiver coerente. Por outro lado e mesmo estando o DFD incoerente a informação é registada nas matrizes de modelação gráfca do sistema, isto foi pensado para que possam existir gravações intercalares no projecto isto é, gravações temporárias sem que o projecto esteja completo e coerente. Porém a cada gravação o utilizador é informado das falhas que foram detectadas ao testar a coerência do projecto.

NOTA IMPORTANTE : Ao alterar o Desenho do projecto é necessário sincronizar novamente a base de dados para que as alterações tenham efeito. O procedimento de sincronização está explicado mais adiante neste manual.

3.2. Ambiente de Trabalho

Neste capítulo irão ser descritos em pormenor todos os aspectos que têm a ver com o ambiente gráfico de trabalho onde se irá trabalhar na modelação dos Diagramas de Fluxos de Dados no `SysObj`.

Aqui estão incluídas as explicações de todos os botões das barras de ferramentas bem como o que significam todos os símbolos e ícones utilizados.

3.2.1. Barras de Ferramentas

A interface do `SysObj` é composta por 4 áreas distintas (Figura 2.1) a barra de ferramentas do projecto (1), a barra de ferramentas da implementação gráfica (2), a lista de "objectos" existentes (3) e a área de implementação composta por várias janelas de implementação (4). Existe ainda um menu de propriedades dos objectos desenhados (5).

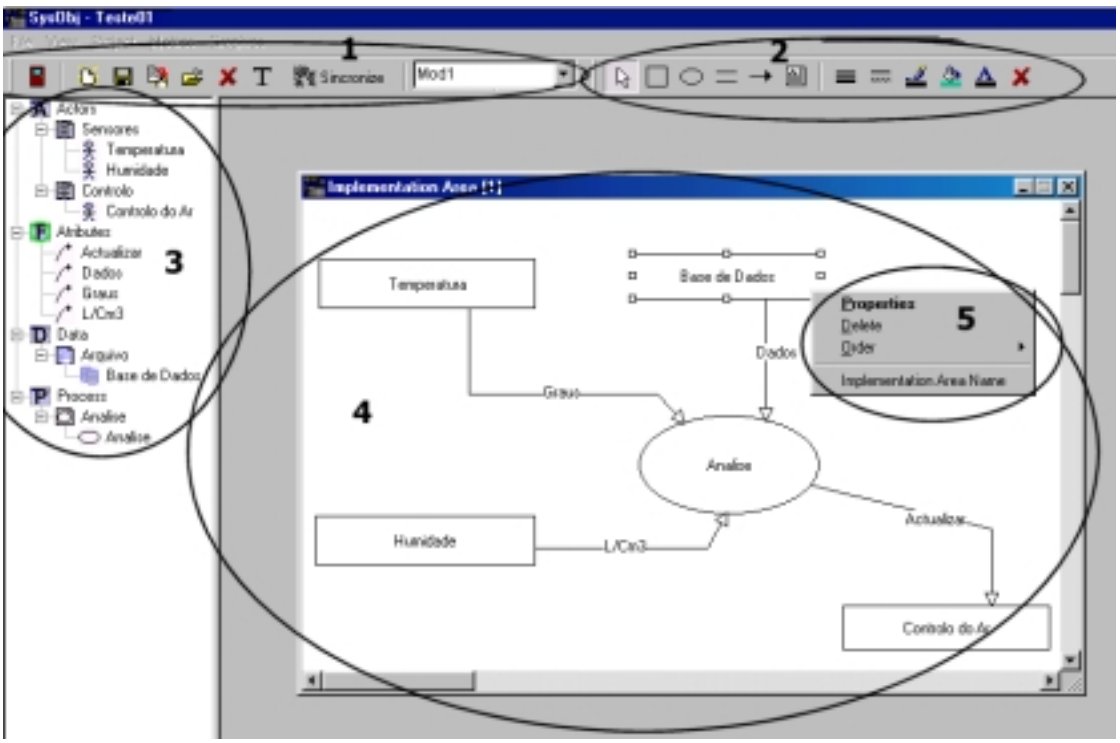

Figura 3.2.1 – Áreas distintas da interface

3.2.1.1. Barra de Ferramentas do Projecto

Figura 3.2.2 – Barra de ferramentas do projecto

Figura 3.2.3 – Sair

Sai da Interface Gráfica, se foram efectuadas alterações ao projecto é perguntado ao utilizador se deseja gravar as alterações (Figura 3.2.4).

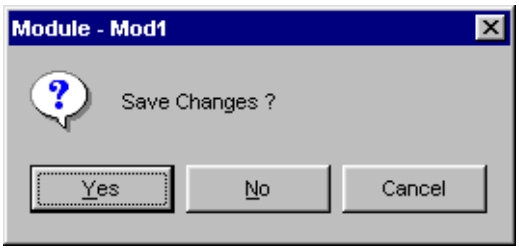

Figura 3.2.4 – Guardar alterações

Figura 3.2.5 – Nova área de implementação

Em termos de programa é criada uma nova área de implementação com o nome : "Implementation Area [xx]" onde xx é o número de janelas existentes. Podem ser criadas tantas janelas quanto se pretender, sendo o único limite para o número de janelas a memória e o espaço em disco disponível do computador utilizado. Em termos práticos é criado um novo "*use case*".

Figura 3.2.6 – Gravar

Grava as alterações feitas ao projecto. **ATENÇÃO**, gravar não sincroniza as bases de dados, apenas guarda a implementação do projecto para futuras alterações. Assim, "Gravar" não altera as matrizes do `SysObj` nem invalida as métricas calculadas. É efectuado o teste de coerência ao DFD, porém este (no caso de ser negativo – DFD Incoerente) não tem qualquer efeito sobre a gravação, ou seja, é possível Gravar DFD's incoerentes. Esta opção de gravação só está disponível quando há alterações feitas no projecto.

Figura 3.2.7 – Copiar para a área de transferência

Copia o desenho (implementação) da janela activa para o *ClipBoard* (ou Área de Trabalho), para posterior utilização num programa exterior ao `SysObj`, por exemplo para incluir imagens das implementações num documento de texto ou mesmo numa apresentação.

Figura 3.2.8 – Anular

Anula as últimas alterações feitas. Esta opção repõe a última gravação feita ao módulo corrente. A opção de anulação só está disponível quando há alterações feitas no projecto. Ao fazer Anular é perguntado ao utilizador se deseja mesmo repor a última gravação efectuada ao DFD (Figura 3.2.9).

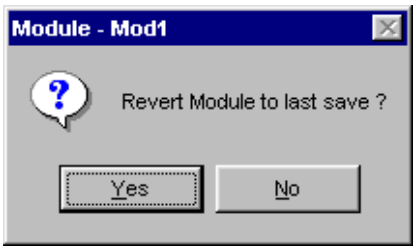

Figura 3.2.9 – Repor última gravação

Figura 3.2.10 – Visualizar áreas de implementação

Permite visualizar as áreas de implementação existentes e os atributos de entrada e saída que cada uma tem (Ver Visualização das Áreas de Implementação na página : 23). Em termos práticos permite-nos ver todos os *"use cases"* existentes no projecto e as ligações entre eles.

Figura 3.2.11 – Apagar área de implementação

Apaga uma área de implementação. **ATENÇÃO**, ao apagar uma área de implementação todos os objectos que a mesma contém serão eliminados. O utilizador é confrontado com a seguinte mensagem de aviso (Figura 3.2.12).

Figura 3.2.12 – Apagar objectos existentes

Podem ainda ser apresentadas mensagens relativas á eliminação de determinados objectos (Ver Eliminar Objectos na página : 33).

Figura 3.2.13 – Teste de coerência

Testa a integridade do Diagrama de Fluxos de Dados do Projecto. Um DFD para ser coerente tem que obedecer ás seguintes regras:

- Não podem existir objectos sem atributos
- Não podem existir "classes" de objectos sem atributos de entrada e de saída, isto é, agrupando as várias implementações do mesmo objecto têm que existir sempre atributos de entrada e de saída.

As mensagens de erro, no caso do DFD estar incoerente, referem especificamente qual o objecto ou classe de objectos que não está correcto e porque motivo.

Figura 3.2.14 – Sincronizar bases de dados

Sincroniza a base de dados do DFD com a do `SysObj`. É efectuado um teste de integridade ao DFD antes da sincronização. A sincronização só é efectuada se o DFD estiver coerente. Após a sincronização das bases de dados as métricas calculadas ficam invalidadas uma vez que existem dados novos nas matrizes do `SysObj`.

Figura 3.2.15 – Alternar entre módulos do projecto

Permite alternar entre os módulos existentes do projecto, se foram efectuadas alterações ao modulo corrente é perguntado se deseja gravar as alterações (Figura 3.2.16).

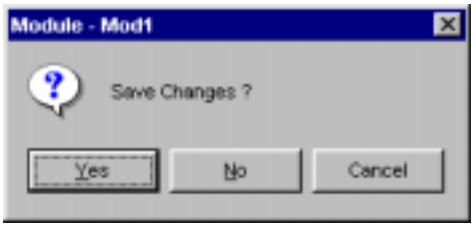

Figura 3.2.16 – Gravar alterações

3.2.1.2. Barra de Ferramentas da Implementação Gráfica

Figura 3.2.17 – Barra de ferramentas da implementação gráfica

Figura 3.2.18 – Cursor

Cursor, quando este botão está activo o cursor do rato pode ser utilizado para seleccionar, mover e redimensionar objectos (Ver Selecção de Objectos na página 30).

Figura 3.2.19 – Actor

Actor, quando este botão está activo o Cursor do rato é utilizado para desenhar Actores (Ver Desenho de Objectos na página 28).

Figura 3.2.20 – Processo

Processo: quando este botão está activo o Cursor do rato é utilizado para desenhar Processos (Ver Desenho de Objectos na página 28).

Figura 3.2.21 – Arquivo

Arquivo: quando este botão está activo o Cursor do rato é utilizado para desenhar Arquivos (Ver Desenho de Objectos na página 28).

Figura 3.2.22 – Atributo

Atributo: quando este botão está activo o Cursor do rato é utilizado para desenhar Atributos (Ver Desenho de Objectos na página 28).

Figura 3.2.23 – Texto

Texto: quando este botão está activo o Cursor do rato é utilizado para desenhar caixas de Texto (Ver Desenho de Objectos na página 28).

Figura 3.2.24 – Espessura da linha

Se estiver algum objecto seleccionado este botão altera a espessura da linha. Se não estiver nenhum objecto seleccionado todas as linhas desenhadas irão ter a espessura aqui especificada. Esta propriedade dos objectos é meramente visual e orientadora para o projectista, não tem quaisquer implicações a nível das matrizes do `SysObj`.

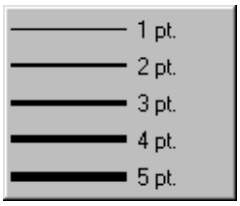

Figura 3.2.25 – Diferentes espessuras de linha

Figura 3.2.26 – Estilo da linha

Estilo da Linha, se estiver algum objecto seleccionado este botão altera o estilo da linha (contínua, tracejada, etc.). Se não estiver nenhum objecto seleccionado todas as linhas desenhadas irão ter o estilo aqui especificado.

Figura 2.27 – Diferentes estilos de linha

ATENÇÃO : no caso dos atributos qualquer estilo de linha que não seja contínua implica que esse atributo seja um atributo de controlo.
NOTA : Os estilos de linha diferentes de contínua só serão visíveis se a espessura da linha for 1.

Figura 3.2.28 – Cor da linha

Cor da Linha: se estiver algum objecto seleccionado este botão altera a cor da linha. Se não estiver nenhum objecto seleccionado todas as linhas desenhadas irão ter a cor aqui especificada. Esta propriedade dos objectos é meramente visual e orientadora para o projectista, não tem quaisquer implicações a nível das matrizes do `SysObj`.

Figura 3.2.29 – Diferentes cores de linha

Figura 3.2.30 – Cor de fundo

Cor de Fundo: se estiver algum objecto seleccionado, este botão altera a cor do Fundo. Se não estiver nenhum objecto seleccionado todos os objectos desenhados irão ter o fundo com a cor que aqui for especificada. Esta propriedade dos objectos é meramente visual e orientadora para o projectista, não tem quaisquer implicações a nível das matrizes do `SysObj`.

Figura 3.2.31 – Diferentes cores de fundo

Figura 2.32 – Estilo e cor do texto

Estilo e Cor do Texto: se estiver algum objecto seleccionado este botão permite alterar o estilo e cor do texto. Se não estiver nenhum objecto seleccionado todos os objectos desenhados irão ter a cor e o estilo de texto que aqui forem especificados. Esta propriedade dos objectos é meramente visual e orientadora para o projectista, não tem quaisquer implicações a nível das matrizes do `SysObj`.

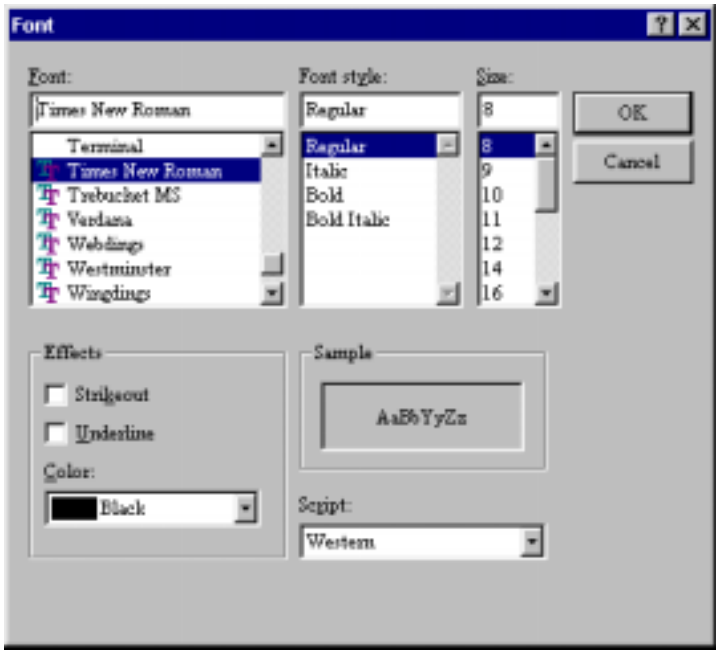

Figura 2.33 – Diferentes estilos e cores do texto

Figura 2.34 – Apagar objecto seleccionado

Apagar, este botão apaga o objecto que estiver seleccionado. Podem ainda ser apresentadas mensagens relativas á eliminação de determinados objectos (Ver Eliminar Objectos na página: 33).

3.2.2. Lista de Objectos Existentes

Figura 3.2.35 – Lista de objectos existentes

A lista de Objectos encontra-se organizada da seguinte maneira :

Figura 3.2.36 – Actores

Actores. No nível seguinte estão as classes de Actores e dentro de cada classe estão as diferentes implementações da mesma .

Figura 3.2.37 – Atributos

Atributos. No nível seguinte estão todos os atributos existentes .

Figura 3.2.38 – Arquivos

Arquivos. No nível seguinte estão as classes de Arquivos e dentro de cada classe estão as diferentes implementações da mesma .

Figura 3.2.39 – Processos

Processos. No nível seguinte estão as classes de Processos e dentro de cada classe estão as diferentes implementações da mesma .

Figura 3.2.40 – Agrupamento de objectos seleccionado

O circulo verde á volta dos símbolos de cada agrupamento de objectos indica que esse agrupamento está seleccionado.

Através desta lista de objectos também é possível seleccionar objectos bastando para isso fazer *click* na implementação do objecto que se quer seleccionar. A área de implementação onde se encontra o objecto seleccionado passa a ser a área activa.

Ao fazer *click* com o botão direito do rato acede-se ao menu de atalhos (Ver Menu de Atalhos na página 25)

3.2.3. Área de Implementação

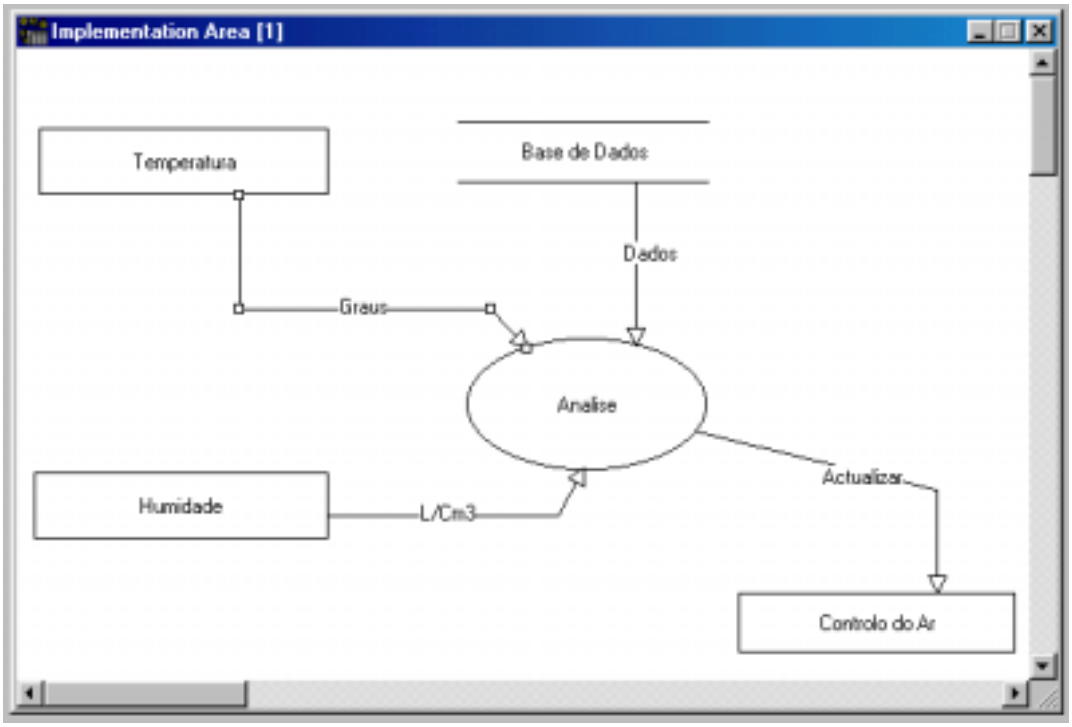

Figura 3.2.41 – Área de implementação

É a área utilizada para implementar graficamente os diagramas de fluxos de dados (Ver Desenhar Objectos na página 28). Ao fazer *click* dentro desta área com o botão do lado direito do rato acede-se ao menu de atalhos (Ver Menu de Atalhos na página : 25).

Ao fazer Duplo *click* sobre um objecto na área de implementação acede-se á janela de propriedades do objecto (Ver Menu de Atalhos na página : 25).

3.2.4. Visualização das Áreas de Implementação

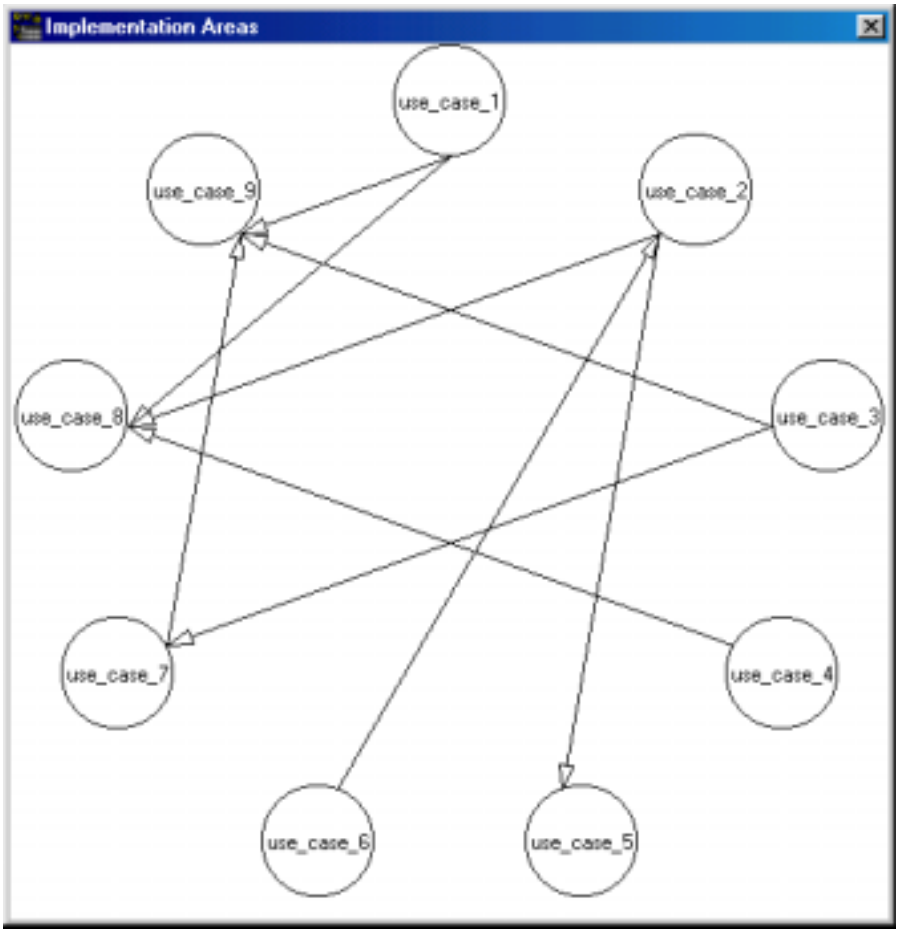

Figura 2.42 – Visualização das áreas de implementação

Aqui é possível ver todas as áreas de implementação existentes e a forma como estas se interligam(ou não) através de atributos. O sentido da seta indica se o conjunto de atributos são de entrada ou de saída na área de implementação.

Ao fazer *click* com o botão direito do rato numa circunferência é mostrada uma mensagem que lista todos os objectos, actores ou arquivos, que entram e saem da área de implementação correspondente:

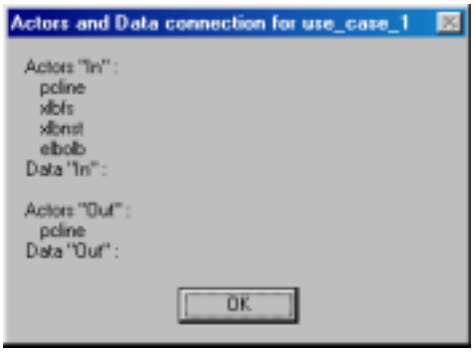

Figura 3.2.43 – Lista de objectos ligados

- *'Actors In'* - todos os actores que estão na área de implementação correspondente com atributos de entrada.

- *'Actors Out'* - todos os actores que estão na área de implementação correspondente com atributos de saída.

- *'Data In'* - todos os arquivos que estão na área de implementação correspondente com atributos de entrada.

- *'Data Out'* - todos os arquivos que estão na área de implementação correspondente com atributos de saída.

Ao fazer duplo *click* numa seta que interliga duas áreas de implementação é mostrada uma lista dos atributos que entram ou saem de uma área de implementação para a outra (Figura 3.2.44).

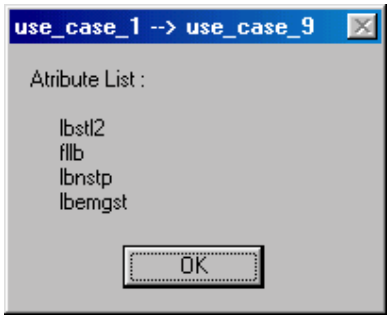

Figura 3.2.44 – Lista de atributos

Ao fazer duplo *click* numa circunferência a janela é fechada e é activada a área de implementação correspondente.

Para maior e melhor visualização dos *"use cases"* mesmo nos projectos mais complexos, esta janela pode ser redimensionada, as circunferências correspondentes ás áreas de implementação e as setas que os interligam ajustam-se ao tamanho da janela.

3.2.5. Menu de Atalhos

Figura 3.2.45 – Menu de atalhos

Estes menus podem ser acedidos de duas maneiras:

- Fazendo *click* com o botão do lado direito do rato sobre um objecto na lista de objectos. **NOTA :** Se não estiver nenhum objecto seleccionado só aparece a opção *Implementation Area Name.*

- Ao fazer *click* com o botão do lado direito do rato na área de implementação. **NOTA :** Se não estiver nenhum objecto seleccionado só aparece a opção *Implementation Area Name*.

***Properties* - Propriedades do Objecto** dá acesso a uma janela em que se pode alterar o nome e a classe dos objectos. No caso do objecto ser um atributo é possível ainda visualizar os objectos de origem e destino bem como o tipo de atributo, isto é, se é um atributo de dados ou de controlo. A janela de propriedades pode ainda ser acedida através de duplo *click* na área de implementação sobre um objecto e através de duplo *click* sobre uma qualquer implementação de um objecto na lista de objectos

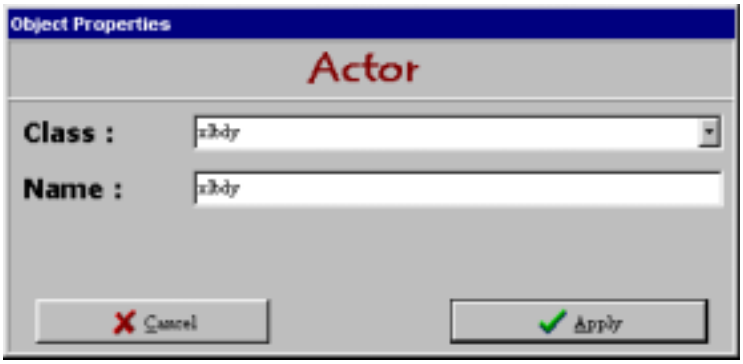

Figura 3.2.46 – Propriedades de um objecto

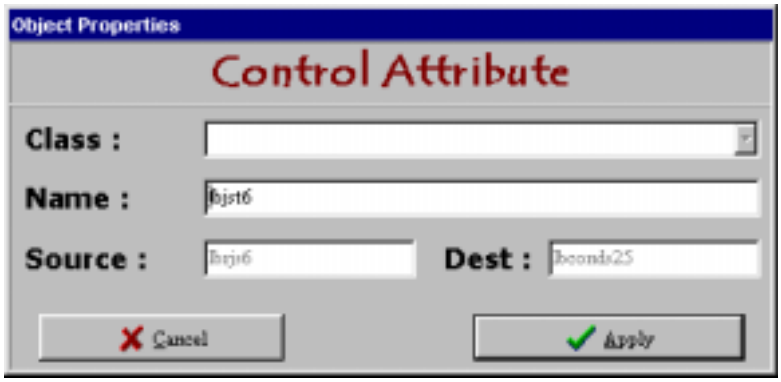

Figura 3.2.47 – Propriedades de um atributo

***Delete* - Apagar objecto** apaga o objecto que está seleccionado. Podem ainda ser apresentadas mensagens relativas á eliminação de determinados objectos (Ver Eliminar Objectos página : 33).

***Implementation Area Name* - Nome da área de implementação** abre uma janela que permite alterar o nome da área de implementação que está activa.

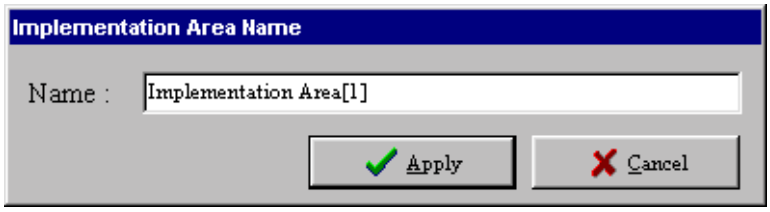

Figura 3.2.48 – Nome da área de desenho/implementação

3.3. Métodos de Trabalho

Cada tipo de objecto tem características particulares tais como a forma de desenhar, o tipo de comportamento quando movido ou redimensionado e a forma como deve ser seleccionado. Os objectos reagem ainda de forma diferente á eliminação, existem objectos "dependentes", objectos que têm ligações através de atributos a outros objectos, e ainda objectos que têm várias implementações dentro da mesma classe.

Este capítulo tenta então abordar todos esses aspectos para que o utilizador aproveite todas as funcionalidades que a interface gráfica do `SysObj` proporciona.

3.3.1. Desenho de Objectos

Actores, Arquivos e Processos :

Seleccionar o botão correspondente ao objecto que se deseja seleccionar. Em seguida fazer *click* com o rato na área de implementação, este ponto irá corresponder ao canto superior esquerdo do objecto, depois basta arrastar o rato (mantendo pressionado o botão) e soltar o botão do rato quando o objecto tiver o tamanho desejado. Posteriormente pode ainda alterar as dimensões do objecto bem como a sua posição na área de implementação (Ver Mover/Redimensionar objectos na página : 31). O programa ajusta o tamanho do objecto para um mínimo de 50 pixels horizontais e verticais. É atribuído ao objecto uma classe e um nome de implementação por defeito. O Nome do objecto e a classe a que ele pertence podem ser posteriormente alterados (Ver Menus de Atalho na página: 25).

Texto Livre :

Fazer *click* com o rato na área de implementação, este ponto irá corresponder ao canto superior esquerdo do objecto, depois basta arrastar o rato (mantendo pressionado o

botão) e soltar o botão do rato quando o objecto tiver o tamanho desejado. Posteriormente pode ainda alterar as dimensões do objecto bem como a sua posição na área de implementação (Ver Mover/Redimensionar objectos na página : 31). O programa ajusta o tamanho da caixa de texto ao tamanho do nome introduzido, tendo em conta o tipo e tamanho da fonte escolhida para esse objecto. É atribuído ao objecto um nome por defeito. O Nome do objecto pode ser posteriormente alterado (Ver Menus de Atalho na página: 25).

Atributos :

Começar por seleccionar o botão correspondente ao desenho de atributos. Fazer *click* com o rato dentro de um objecto já existente na área de desenho, este objecto irá ser o objecto de origem. Sem manter o botão do rato pressionado mover o rato e fazer *click* na área de implementação dentro ou fora de um outro objecto existente. Se o *click* for feito fora de algum objecto corresponde á criação de mais um ponto no desenho do atributo. O número de pontos que se pode criar por atributos é ilimitado. Se o *click* for efectuado dentro de algum objecto corresponde á definição desse objecto como objecto de destino do atributo que foi criado. O programa ajusta o tamanho do primeiro e último segmento de recta para o limite dos objectos escolhidos, isto para que as linhas fiquem coincidentes.

NOTA : Se a linha do atributo for diferente de contínua indica que é um atributo de Controlo, se a linha for contínua significa que é um atributo de dados.

O Nome do atributo é gerado automaticamente da seguinte forma :
“Nome do processo de origem” + “->” + **“Nome do processo de destino”**. O nome do objecto bem como a classe a que ele pertence, podem posteriormente ser alterados (Ver Menus de Atalho na página : 25). O nome criado é colocado no centro da média de todos os pontos que constituem o atributo, esse nome pode posteriormente ser movido (Ver Mover/Redimensionar objectos na página : 31).

3.3.2. Selecção de Objectos

Um objecto pode ser seleccionado fazendo *click* no objecto na área de implementação ou fazendo *click* no objecto na lista de objectos. No caso dos atributos o objecto pode ser seleccionado fazendo *click* no texto ou nos segmentos do atributo.

3.3.3. Mover / Redimensionar Objectos

Para mover ou redimensionar um objecto este tem que estar previamente seleccionado. Depois de ter o objecto seleccionado podem surgir três situações dependendo do tipo de objecto:

Actores, Arquivos e Processos :

Para mover um objecto deste tipo basta arrastá-lo (Mover mantendo o botão do rato pressionado) para a posição desejada, tendo em conta que os atributos desse objecto se ajustam á mudança de posição. Para redimensionar basta arrastar na direcção desejada um dos 8 pontos de redimensionamento do objecto.

Caixas de Texto :

Para mover um objecto deste tipo basta arrastá-lo (Mover mantendo o botão do rato pressionado) para a posição desejada. Estes objectos não são redimensionáveis, o tamanho é ajustado ao nome do objecto, tendo em conta o tipo de fonte e o tamanho.

Atributos :

Ao arrastar (Mover mantendo o botão do rato pressionado) um atributo só os pontos/segmentos interiores é que são arrastados, os pontos de origem/destino ficam fixos nas extremidades dos objectos. Isto serve para garantir que as extremidades do atributo não fiquem fora dos objectos de origem/destino.

Os segmentos dos atributos podem ser redimensionados/movidos, bastando para isso arrastar um dos pontos de redimensionamento existentes. Os restantes segmentos do atributo manter-se-ão constantes.

Os textos dos atributos são independentes dos segmentos, isto é, ao mover um atributo a posição da caixa de texto do atributo não é movida. Para mover o texto do atributo proceder da mesma forma que se procederia para uma caixa de texto normal.

3.3.4. Eliminar Objectos

Ao eliminar um objecto seleccionado o programa analisa as seguintes situações :

- A classe do objecto a eliminar tem mais implementações ?
- O objecto tem atributos de entrada ou de saída ?

Se a classe do objecto a eliminar tiver mais implementações é perguntado ao utilizador se deseja também eliminar as outras implementações dessa classe. Essas implementações podem ou não estar na mesma área de implementação.

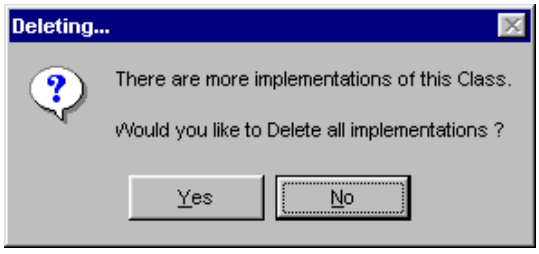

Figura 3.3.1 – Apagar todas as implementações do objecto

Se o objecto está ligado por atributos de entrada ou de saída é informado ao utilizador que esses atributos também serão eliminados, uma vez que não podem existir atributos que não tenham objecto de origem ou de destino.

Figura 3.3.2 – Apagar atributos do objecto

De notar que este procedimento é recursivo. Por exemplo se vai ser eliminado um objecto que tem mais implementações mas que não tem atributos, é questionado ao utilizador se quer eliminar o resto das implementações dessa classe. Se a resposta for positiva é possível ir de encontro á segunda situação uma vez que os outros objectos que irão ser eliminados podem ter atributos de entrada ou saída.

Bibliografia

[EP98] Hans-Erik Erikson and Magnus Penker, “UML Toolkit”, Jonh Wiley & Sons, Inc., 1998

[FS97] Martin Fowler with Kendal Scott, “UML Distilled – Applying the Standard Object Modeling Language”, Addison-Wesley Longman, Inc., 1997

[R97a] Rui M. V. Ribeiro, “Sistema Electrónico para Automatização de Linha de Produção Vinícola”, Tese de Mestrado em Engenharia Electrónica e de Computadores, IST, Outubro, 1997.

[RDTT98] R Ribeiro, O. P. Dias, J. P. Teixeira and I.C. Teixeira, “Automation and Real-Time Control of a Wine Bottling Production Line – And Object-Oriented Based Hardware/Software Co-Design Case Study”, Preprints of the 8th IFAC/IFORS/IMACS/IFIP Symposium on Large Scale Systems : Theory & Applications, vol. II, pp. 1211-1216, July, 1998.

[RJB99] James Rumbaug, Ivar Jacobson and Grady Booch, “The Unified Modeling Language Reference Manual”, Addyson- Wesley Longman, Inc., 1999.

4. SysObj *v4.0* – Manual Técnico

Resumo

A versão 4.0 do `SysObj` foi desenvolvida com o objectivo de criar uma interface gráfica para o desenho de Diagramas de Fluxo de Dados (DFDs), diagramas esses que são utilizados para modelar e descrever as funcionalidades do projecto, dando assim ao projectista uma total independência das ferramentas CAD para o desenvolvimento dos mesmos DFDs.

A versão anterior do SysObj foi desenvolvida para aceitar DFDs previamente desenhados na ferramenta `ParadigmPlus`. No `ParadigmPlus` o projectista desenha os DFDs e exporta-os para ficheiros '*.cdf', os quais eram posteriormente importados no `SysObj` a fim de serem processados.

Na presente versão do `SysObj` o projectista pode realizar toda a modelação do projecto dentro do ambiente da ferramenta, incluíndo os DFDs.

Notação

Neste relatório foram utilizados diferentes estilos gráficos, de forma a facilitar a interpretação dos vários conceitos. A notação utilizada foi a seguinte :

Expressões em inglês estão apresentadas em *itálico*
Conceitos que se pretendem destacar são apresentados em bold
Nomes de ferramentas estão representados em `Courier New`
O restante texto está apresentado em Times New Roman

Lista de Figuras

4.1.1 – Representação de um sistema no diagrama *use-case* 39
4.1.2 – Símbolos usados para representar actores em UML 40
4.1.3 - Símbolos para o *use-case* 40
4.1.4 – Relações comunicação, extensão e utilização 41
4.1.5 – Diagrama *use-case* para um conjunto de funcionalidades específicas de um equipamento de electrocardiograma (ECG) 42
4.1.6 – Realização de um *use-case* com a colaboração de três classes 43
4.1.7 – Diagrama de Contexto do `SysObj` 44

4.3.1 – *Use-case* de Sincronização.....48
4.3.2 – Diagrama de sequência do *use-case* de sincronização 49
4.3.3 – Sincroniza Matrizes com DFD.....50
4.3.4 – *Use-case* do desenho de actores, processos, arquivos e textos 52
4.3.5 – Diagrama de sequência do *use-case* de desenho de objectos ...53
4.3.6 – *Use-case* do desenho de um atributo.....54
4.3.7 – Diagrama de sequência do *use-case* de desenho de um atributo 56

4.4.1 – Implantação das linhas de enchimento para vinhos e espumantes [R97a] 58
4.4.2 – Diagrama de alto nível dos módulos do sistema fill [R97a].....60
4.4.3 – Sub-blocos do sistema de controlo (SEMi) para cada máquina [R97a] 61
4.4.4 – Disposição dos sensores [R97a].....61
4.4.5 – Diagrama de estados da linha [R97a] 63
4.4.6 - Menu de criação de projecto.....63
4.4.7 – Criação do projecto 64
4.4.8 – Menu da interface gráfica.....64
4.4.9 – Botão de criação de uma área de implementação.....65
4.4.10 – Opção de mudar o nome da área de implementação 65
4.4.11 – Alteração do nome da área de implementação.....65
4.4.12 – Diagrama do *use-case* "paragem da máquina".....66
4.4.13 – Botão de desenho de actores 67
4.4.14 – Criação do actor.....67
4.4.15 – Objecto criado com nome por defeito.....67
4.4.16 – Botão de selecção de objectos 67
4.4.17 – Objecto seleccionado.....68
4.4.18 – Menu de atalhos do objecto 68
4.4.19 – Propriedades do objecto 68
4.4.20 – Botão de desenho de processos.....69
4.4.21 – Botão de desenho de atributos 69
4.4.22 – Criação de um atributo.....69

4.4.23 – Botão de selecção do tipo de linha.....70
4.4.24 – Atributo de controlo desenhado 70
4.4.25 – Diagrama do *use-case* "máquina em produção" 71
4.4.26 – Diagrama do *use-case* "configuração da máquina".....72
4.4.27 – Diagrama do *use-case* "correcção da velocidade versus congestionamento na saída" 73
4.4.28 – Diagrama do *use-case* "carga de rótulos" 74
4.4.29 – Diagrama do *use-case* "actualização do estado dos sensores" 75
4.4.30 – Diagrama do *use-case* "informação para o estado da linha".....76
4.4.31 – Diagrama do *use-case* "aviso de variação de velocidade".....76
4.4.32 – Diagrama do *use-case* "aviso da situação da máquina" 77
4.4.33 – *Autonomia* média total (ATAt) para o conjunto de arquitecturas possível 83
4.4.34 – Número de malhas (nlt) para o conjunto de arquitecturas possível 83
4.4.35 – Coesão média total (ATCoh) para o conjunto de arquitecturas possível 84
4.4.36 – Acoplamento total do sistema (ASTCp) para o conjunto de arquitecturas possível 84

Lista de Tabelas

4.2.1 – Objectos 46
4.2.2 – Nós dos atributos 47

4.4.1 – Modelo Funcional da UP do SEM 79
4.4.2 – Dicionário dos processos e atributos do DFD da UP do SEM.....81
4.4.3 – Dicionário dos actores que interagem com a UP do SEM.....82
4.4.4 – Valores das métricas de desempenho 84
4.4.5 – Valores das métricas de testabilidade intrínseca e desvios padrão 85
4.4.6 – Lista dos agentes de origem e destino do SEM 87
4.4.7 – Lista de colaboração para a realização dos *use-cases* 89
4.4.8 – Características da arquitectura-alvo.....90

Glossário

Acção – Interacção de um actor num processo com o objectivo de produzir um resultado. Os actores só podem agir ou realizar acções em processos. Esta **acção** é realizada através de atributos.

«Actor» - Simbologia utilizada para representar um actor externo num diagrama de *use-cases*. Um actor externo é um actor que não pertence ao *use-case* em causa mas que interage com ele.

Actor (*Actor*) – Entidades que interagem com os processos, activam processos, recolhem informações dos processos, etc. Em sistemas industriais os actores podem ser vistos como sendo os operadores das máquinas.

Área de Implementação (*Implementation area*) – Área onde são implementados/desenhados os objectos. Estas áreas podem ser utilizadas para criar *use cases* separados, com diferentes níveis de detalhe. Por exemplo numa primeira área eu utilizo um processo da classe "xpto" e numa área distinta vou então descrever o processo "xpto" com mais detalhe, ou seja vou ter uma nova implementação da classe "xpto" que tem mais atributos, mais consultas a dados, etc. Esta separação é apenas lógica e só serve para ajudar o projectista na esquematização do projecto uma vez que o DFD é global.

Atributo (*Atribute*) – Podem ser atributos de entrada ou saída dos processos. Os atributos são as ligações dos processos a actores ou a bancos de dados transportam, ou dados no caso de serem atributos de dados (*Data Atributes*), ou acções, ordens de execução, no caso de serem atributos de controlo (*Control Atributes*). Um objecto do tipo atributo não tem agrupamento em classes uma vez que não fazia sentido ter várias implementações de um mesmo fluxo de dados ou de controlo.

Classe (*Class*) – Agrupamento de Objectos, uma classe agrupa uma ou mais implementações de objectos. Por exemplo se eu tiver uma classe de Actores chamada "Sensores" eu posso ter várias implementações do Actor "Sensores" que podem ser, "Sensor Temperatura", "Sensor de Humidade", etc. Estes dois actores são dois objectos que representam implementações diferentes da classe "Sensores" e como tal tem atributos diferentes e entram (ou não) em processos diferentes.

Dados (*Data*) – Bancos de dados, estes objectos guardam informação necessária a actores ou a processos. Estas informações podem ser obtidas por actores e processos e actualizadas através dos processos, sempre através de atributos.

DFD – Diagrama de Fluxos de Dados (Data Flow Diagram), servem para descrever funcionalidades. Tomando o exemplo anterior pode ser definido um

DFD para o *use case* de medir temperaturas. Os DFDs têm vários níveis de abstracção, isto é, começando com o DFD de nível 0, nível mais abstracto, que descreve uma funcionalidade pode-se ir detalhando cada funcionalidade nova que vai surgindo num outro DFD de nível mais baixo. Estas funcionalidades novas vão surgindo quando se está a detalhar uma outra funcionalidade de nível de abstracção superior. Assim pode-se equiparar um *use case* a um DFD de nível 0 porque têm níveis de abstracção equivalentes.

DFD coerente – Para um DFD estar coerente têm que estar cumpridas as seguintes condições :

- Cada objecto tem que ter pelo menos um atributo, seja um atributo de entrada ou de saída;
- Cada classe tem que ter atributos de entrada e de saída (Tendo em conta todos os objectos dessa classe).

***Handle* –** Valor que indica uma posição de memória onde existe um objecto. No caso o objecto pretendido é a área de desenho onde a aplicação vai desenhar os objectos com base nos *inputs* do rato.

Instância – Espaço temporal em que alguém, que é utilizador e pré-definido do sistema, interage com o mesmo.

Objecto (*Object*) – Define a implementação de uma figura no diagrama de fluxo de dados, este objecto pertence a uma dada classe, podendo existir mais objectos que sejam outras implementações da mesma classe. Um objecto pode ser, um Actor (*Actor*), um Processo (*Process*) ou um Atributo (*Atribute*), se bem que neste último caso (atributos) não existe a noção de classe.

Processo (*Process*) – Processam informação através de dados fornecidos por actores ou por bancos de dados. Os atributos de controlo a partir de actores ou de outros processos activam os processos.

Sincronização – Processo de sincronizar as matrizes de desenho da interface gráfica do `SysObj` com as matrizes finais a ultilizar posteriormente na análise do projecto.

***UML (Unified modeling language)* –** Esta linguagem é o resultado do esforço de uniformização das diversas metodologias orientadas para objectos para *hardware* e *software*. A *UML* permite descrever qualquer tipo de sistema em termos de diagramas orientados para objectos.

***Use Case* –** Representa a funcionalidade de uma dada tarefa ou procedimento ou num cenário mais completo, uma das funcionalidades de um sistema complexo. Por exemplo se existir um projecto para medir temperaturas e tomar acções consoante as medidas pode-se considerar a funcionalidade de medição um *use case* e a funcionalidade de tomar acções outro *use case*. Os *use cases* são usualmente as primeiras definições de um projecto quando ainda não exixte uma ideia bem clara sobre como cada

funcionalidade deve ser executada. Isto deve-se ao elevado poder de abstracção dos *use cases*.

4.1. Introdução

Neste texto descrevem-se as estruturas utilizadas na implementação da interface gráfica bem como os procedimentos que manipulam essas estruturas. É descrito em pormenor o procedimento de interligação entre as estruturas de dados da implementação gráfica, ou seja as matrizes da modelação gráfica, e as matrizes do sistema do `SysObj`.

É ainda feito um enquadramento teórico sobre as linguagens de moledação de dados, bem como uma análise às ferramentas já existentes, a fim de fazer o enquadramento do trabalho de implementação da versão 4.0 do `SysObj` assim como a motivação que levou ao desenvolvimento desta interface gráfica.

4.1.1. Enquadramento Teórico

A interface gráfica desenvolvida permite a modelação de projectos com base em diagramas de fluxo de dados que descrevem a funcionalidade modelada pelo *use-case*. Os diagramas *use-case* descrevem o modelo *use-case* do sistema que representa a sua funcionalidade.

Os *use-cases*, recebem e/ou fornecem sempre, os valores de pelo menos um atributo a um actor externo. Um actor é alguém ou outro sistema que por intermédio da troca de mensagens, que são os atributos, interage com o sistema em estudo. Nos diagramas *use-case* o sistema é assumido como uma "caixa preta" que realiza *use-cases*, isto é, funcionalidades, não sendo considerados os aspectos da sua implementação. De facto, usualmente os *use-cases* são identificados na fase inicial do projecto, sem preocupação quanto à sua implementação. Os principais símbolos utilizados nos diagramas *use-case* são: sistema, actores e *use-case* (Figuras 4.1.1(a), (b), 4.1.2(a) (b)). O sistema é representado por um rectângulo com a designação do sistema escrito na sua parte superior, dentro ou fora do rectângulo. No interior do rectângulo ficam localizados os *use-cases*, ou seja as funcionalidades do mesmo, os quais representam a parte focada da funcionalidade total (Figura 4.1.1).

Na interface gráfica desenvolvida os sistemas são representados pelas áreas de implementação onde são desenhadas/especificadas as funcionalidades de cada *use-case* por intermédio de diagramas de fluxo de dados (DFDs).

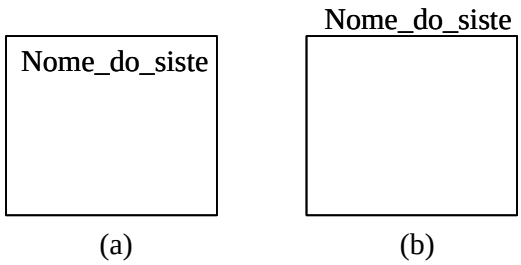

Figura 4.1.1 - Representação de um sistema no diagrama *use-case*

Um actor é alguém ou outro sistema que, por intermédio da troca de mensagens, interage com o sistema em estudo, ou seja "quem" ou "ou quê" usa o sistema. Um actor representa "um papel desempenhado por" e não um utilizador específico do sistema. Por exemplo, na situação "o Carlos Valente quer fazer um seguro de vida na companhia de seguros SempreFixe", é o papel do comprador do seguro e da companhia de seguros nessa aplicação restrita que deve ser modelada e não o indivíduo "Carlos Valente" ou a companhia de seguros "SempreFixe". De facto, uma mesma pessoa pode representar diferentes papéis num sistema, correspondendo deste modo a diferentes actores, que serão modelados em cada uma das situações com o nome do actor a reflectir o papel por ele desempenhado e não a sua funcionalidade ou entidade.

Os actores são representados por intermédio de qualquer dos símbolos representados nas figuras 4.1.2(a) e 4.1.2(b).

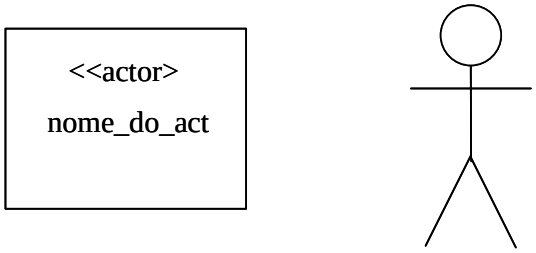

Figura 4.1.2 - Símbolos usados para representar actores em UML

Os *use-cases* ou funcionalidades são representados por elipses com o nome da funcionalidade inscrito no seu interior ou sob elas (Figura 4.1.3(a) e 4.1.3(b)).

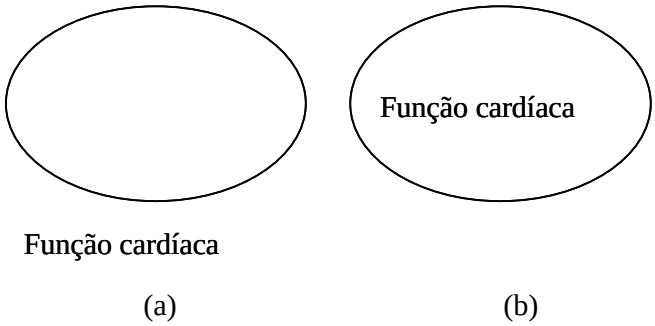

Figura 4.1.3 - Símbolos para o *use-case*

Um *use-case* representa uma funcionalidade completa que é "percebida" por um actor. De facto, a UML utiliza o conceito de *use-case* "como um conjunto de sequências de acções que produz um resultado observável para um actor em particular". Assim,

- um *use-case* é sempre iniciado por um actor. O actor dá directa ou indirectamente ordem ao sistema para executar o *use-case*. Ocasionalmente pode não ser óbvia a acção do actor para iniciar o *use-case*,

- um *use-case* fornece um valor (um resultado) a um dado actor, isto é, um *use-case* deve fornecer algum tipo de valor que é tangível para o utilizador (actor). O valor nem sempre é óbvio, mas deve poder ser visualizado,

- um *use-case* é completo, isto é, um use-case deve ser sempre uma descrição completa.

- Um *use-case* não é uma instância mas uma classe, que descreve a sua funcionalidade como um todo, incluíndo possíveis alternativas, erros e excepções que podem ocorrer durante a sua execução. Uma instância de um *use-case* é designada por cenário e representa uma execução específica do sistema.

A comunicação entre um actor e um *use-case* é representada por uma linha a cheio (Figura 4.1.4) que liga os dois símbolos. Esta associação é designada por associação de comunicação sendo normalmente uma relação ponto-a-ponto, bidireccional, ou seja, as instâncias do actor e do use-case comunicam em ambos os sentidos.

Os *use-cases*, para além de se relacionarem com os actores, podem relacionar-se entre si. Podem identificar-se três tipos de relações entre *use-cases*: extensão (*extends*), utilização (*use*) e agrupamento (*grouping*). As relações de extensão e de utilização são diferentes formas de herança. O agrupamento reúne no mesmo grupo (*package*), *use-cases* relacionados por características comuns. Estas relações entre *use-cases*, exemplificadas nas figuras 4.1.4 e 4.1.5, podem ser definidas do modo seguinte:

- uma relação de extensão (*extend*) de um *use-case* A para um *use-case* B indica que uma instância do *use-case* B pode incluir (sob condições explicitadas na extensão) o comportamento especificado para A,

- uma relação de utilização (*uses*) de um *use-case* A para um *use-case* B uma instância de A também incluí o comportamento especificado para B,

- o agrupamento (*grouping*) consiste em juntar, no mesmo grupo (*package*), *use-cases* com uma funcionalidade similar, ou com alguma relação entre si,

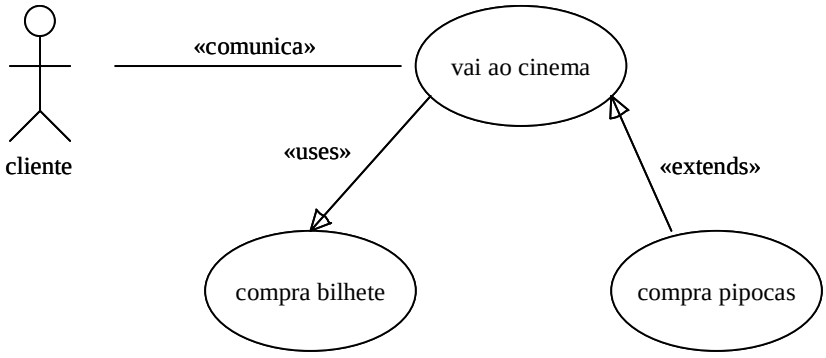

Figura 4.1.4 - Relações comunicação, extensão e utilização

A tarefa de realizar um *use-case* consiste em transformar as diferentes etapas e acções da descrição do *use-case*, feita em texto ou por intermédio de um diagrama de actividades, em classes, operações dentro dessas classes, e relações entre elas, como está representado na figura 4.1.5.

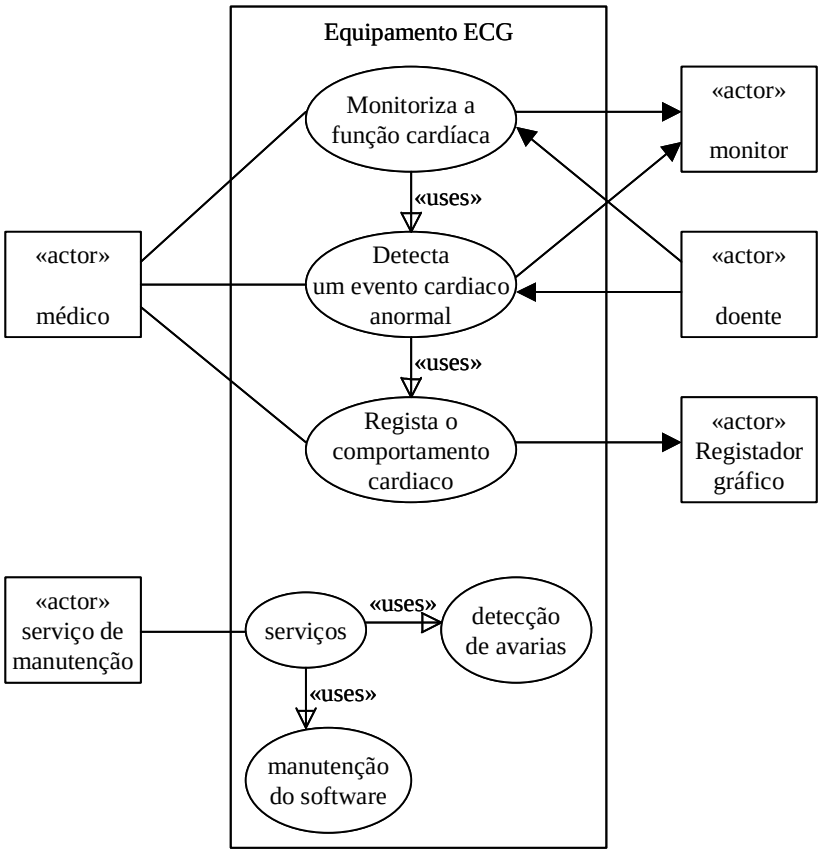

Figura 4.1.5 - Diagrama use-case para um conjunto de funcionalidades específicas de um equipamento de electrocardiograma (ECG)

Deste modo a realização de um *use-case*, consiste em atribuir a responsabilidade de cada funcionalidade de um *use-case* a classes que participam na **colaboração** que o realiza. Essas classes, também designadas por objectos, que participam na colaboração de forma a conseguir a funcionalidade desejada, constituem o **contexto da colaboração**; as interacções que estabelecem entre si constituem a **interacção da colaboração**. Uma colaboração é representada por uma elipse, a traço interrompido, contendo a sua designação. Nesta etapa, a solução encontrada corresponde ao comportamento externo do *use-case*, descrita em termos de uma colaboração dentro do sistema, conforme mostra a figura 4.1.6.

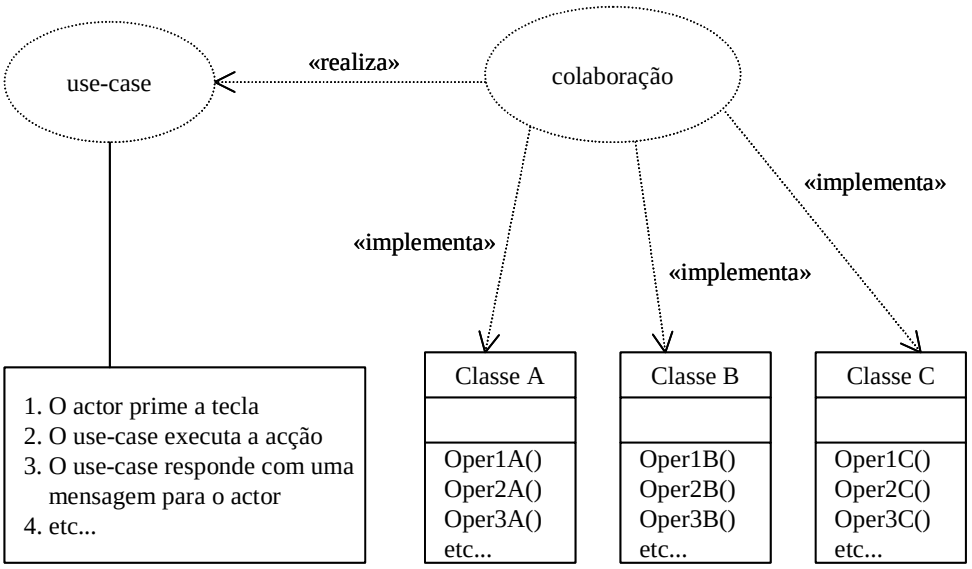

Figura 4.1.6 - Realização de um *use-case* com a colaboração de três classes

Cada funcionalidade do *use-case* é transformada em operações de mais baixo nível, isto é, mais específicas, dentro das classes que participam na colaboração para o realizar. Uma funcionalidade do *use-case* é transformada num dado número de operações, o que sublinha a não existência de uma relação biunívoca entre uma acção no *use-case* e uma operação na interacção entre objectos das classes que participam na colaboração. Note-se que uma classe pode participar em diversos *use-cases*, sendo da responsabilidade das classes fazer a integração dos diferentes papéis que desempenham nos *use-cases*.

A relação entre um *use-case* e a sua implementação, em termos de uma colaboração, é representada por uma **relação de refinamento**, representada na figura 1.6 pela acção «realiza».

Atribuir com sucesso responsabilidades a classes é uma tarefa que requer muita experiência, sendo um trabalho com uma forte componente de interactividade, como aliás sempre acontece quando estão envolvidas metodologias de projecto orientadas para objectos [EP98, FS97, RJB99]. Esta constatação realça a necessidade de se estabelecer, na medida do possível, procedimentos formais que apoiem as equipas de projecto na selecção da arquitectura de classes

4.1.2 Ferramentas Existentes

O SysObj gera automaticamente arquitecturas e faz a avaliação de métricas ou parâmetros de qualidade. Tem como objectivo analisar e organizar em objectos, sistemas electrónicos, tendo sido desenvolvida com a preocupação de ser uma ferramenta flexível, de forma a possibilitar o acréscimo de novas funcionalidades, perturbando o menos possível, as já existentes. Deste modo, utilizou-se uma arquitectura modular, em que os módulos são tão independentes entre si quanto

possível. A própria funcionalidade do SysObj é modelada por diagramas *use-case*, e deste modo, acrescentar funcionalidades à ferramenta corresponde a acrescentar, no modelo, novos *use-case*.

No projecto original, considerou-se como necessária a utilização de diagramas de fluxo de dados, para modelar as funções associadas aos *use-case* que descrevem funcionalidades mais complexas do sistema. O presente projecto adiciona o interface gráfico necessário a esta modulação com DFDs, à ferramenta SysObj.

Esta abordagem permitiu desenvolver esta nova funcionalidade de forma robusta, e completamente integrável com a ferramenta já existente.

O contexto de um sistema representa a fronteira desse sistema com o "mundo" em que se insere. Na figura 4.1.7 representa-se o diagrama de contexto do SysObj, constatando-se que o SysObj dialoga directamente com ferramentas (através de ficheiros e de funções) e com a equipa do projecto (através de ficheiros e da interface gráfica). Ferramentas e utilizadores são assim os actores que enquadram o sistema. A ferramenta *GUI* (*Graphic User Interface*) representa a interface gráfica que foi desenvolvida no SysObj.

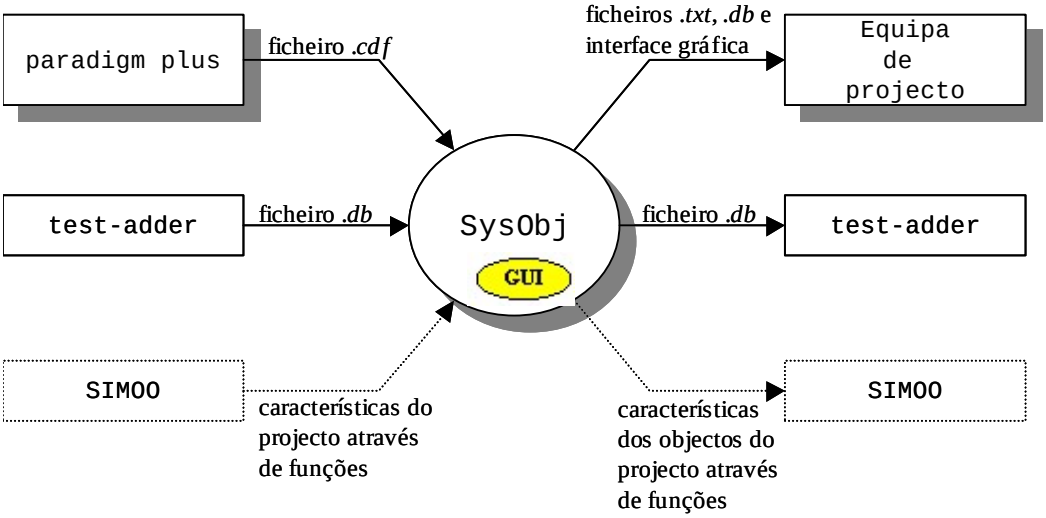

Figura 4.1.7 - Diagrama de Contexto do SysObj

Assim, existem as seguintes entradas e saídas do SysObj :

- O Paradigm Plus é uma ferramenta *CASE* (*Computer Assisted Systems Engeniering*), que de entre outras funcionalidades permite a modelação de um sistema *use-case* em DFDs. Esta ferramenta pode gerar os ficheiros *.cdf (Case Data Interchange) que constituem uma forma de entrada para o SysObj.

- O `test-adder` (ficheiros de formato *.db*) foi desenvolvido de forma compatível de entrada/saída com o `SysObj`. Esta ferramenta é utilizada para adicionar funcionalidades de teste ao projecto em desenvolvimento.

- O `SIMOO` comunica com o `SysObj` por intermédio de funções programadas que contêm toda a informação sobre os objectos do sistema organizados pelo `SysObj`. A partir dessa informação o `SIMOO` gera o diagrama de objectos do sistema, podendo proceder-se á sua simulação.

- O `SysObj` fornece informação à equipa de projecto (ficheiros *.txt*, *.db* e interface gráfica).

4.1.3 Interface Gráfica

Para assegurar um ambiente de trabalho familiar ao utilizador, como o ambiente *windows* ou similares, a interface gráfica realizada neste projecto foi desenvolvida na linguagem de programação `Delphi`, que satisfaz esse requisito.

Para introduzir a nova funcionalidade do `SysObj` foi criada uma estrutura de tabelas adicional para armazenar os dados relativos aos objectos desenhados na interface gráfica. Existem assim, dois momentos na modelação de projecto, o desenho gráfico dos DFDs do projecto e a passagem desses DFDs para as estruturas da primeira versão do `SysObj` onde o projecto é analisado em pormenor.

Foi produzido um CD para a versão 4.0 do `SysObj` que contém as bases de dados de exemplo para dar a conhecer ao utilizador as capacidades desta nova funcionalidade. Exitem duas bases de dados, ou seja dois projectos, a "*fill_1*" e a "*fill_2*". A *"fill_1"* foi gerada pela importação do ficheiro *fill.cdf*(também incluído no CD) criado com a ferramenta *Paradigm Plus*, a *"fill_2"* foi gerada pela nova versão do SysObj no módulo de Implementação Gráfica.

Os Diagramas de Fluxo de Dados apresentados neste trabalho foram desenhados na interface gráfica do SysObj para dar a conhecer as potencialidades da mesma, estando essas bases de dados dos DFDs também incluídas neste CD.

No presente texto a designação `SysObj` refere-se sempre á sua versão 4.0, que incluí a interface gráfica, a qual constitui o projecto descrito e desenvolvido neste trabalho.

4.2. Estruturas de Dados

Neste capítulo estão descritos os aspectos relativos às estruturas de dados utilizadas. Apenas estão incluídas as estruturas da dados físicas uma vez que foi considerado irrelevante para a esquematização técnica do projecto as estruturas temporárias criadas em memória.

4.2.1. Tabelas

Foram criadas as seguintes tabelas adicionais no `SysObj` que armazenam os objectos criados na interface gráfica:

- A tabela de Objectos (OBJECTOS.DB) que guarda todos os objectos criados na interface gráfica associando a cada um toda a informação necessária ao
  seu desenho como se pode ver na tabela 4.2.1.

Nome	Tipo	Tam.	Conteúdo
MODULO	A	15	Nome do módulo a que pertence
ID	I		Número de identificação (único por módulo)
OWNER	A	128	Nome da janela(*use-case*) onde está implementado
STYLE	I		Estilo: Atributo, Actor, Processo, Arquivo
CLASSE	A	200	Classe/Agrupamento
NOME	A	200	Nome
START_X	S		Valor X do ponto de inicial
START_Y	S		Valor Y do ponto de inicial
END_X	S		Valor X do ponto final
END_Y	S		Valor Y do ponto final
PEN_COLOR	I		Cor da linha
PEN_STYLE	S		Estilo da linha: Sólida, Tracejado, Ponteado, etc...
PEN_WITH	S		Largura da linha
BACK_COLOR	I		Cor de fundo
BACK_STYLE	S		Estilo do fundo
FONT_CHARS	S		Charset da fonte
FONT_COLOR	I		Cor da fonte
FONT_HEIGH	S		Altura da fonte
FONT_NAME	A	200	Nome da Fonte
FONT_PITCH	S		Largura das letras da fonte: Normal, Fixa, Variável
FONT_SIZE	S		Tamanho da fonte (em pontos)
FONT_STYLE	S		Estilo fonte: Negrito, Itálico, Sublinhado, Rasurado
ID_ORIGEM	I		No caso de ser um atributo marca o ID de origem
ID_DESTINO	I		No caso de ser um atributo marca o ID de destino

Tabela 4.2.1 – Objectos

- A tabela de Atributos (ATRIBUTOS.DB) que guarda os dados relativos aos nós (pontos de quebra das linhas) de todos os atributos criados na interface gráfica. Os atributos são armazenados nesta tabela, uma vez que, podem necessitar de vários pontos (nós) para serem desenhados. Os atributos estão referenciados na tabela de objectos e os seus nós necessários ao seu desenho estão detalhados na tabela de atributos (Tabela 2.2).

Nome	Tipo	Tam.	Conteúdo
MODULO	A	15	Nome do módulo a que pertence
ID	I		Número de identificação (único por módulo)
N	A	128	Número do nó
X	I		Valor X do nó
Y	A	200	Valor Y do nó

Tabela 4.2.2 – Nós dos atributos

4.3. Descrição de Use-cases

Nesta secção descrevem-se alguns dos *use-cases* mais relevantes para o funcionamento da Interface Gráfica do `SysObj`. Assim, descrevem-se os diagramas de fluxo de dados, as entradas e saídas dos *use-cases*, a descrição detalhada das principais funcionalidades dos *uses-cases*, ou seja os processos que descrevem o *use-case*, assim como os diagramas de sequência, que descrevem em pormenor as acções entre os actores e o ciclo / tempo de vida de cada acção ou actor.

Os diagramas de fluxo de dados, ilustrados neste documento, foram desenhados no `SysObj`, constituindo assim, uma demonstração das características da interface gráfica desenvolvida.

Existe um procedimento/conceito associado á interface gráfica do `SysObj`, designado por **sincronização** que consiste em preencher as matrizes do sistema com os dados que foram introduzidos pelo utilizador sob a forma de desenho dos DFDs.

4.3.1 Interligação entre os DFDs e as Matrizes do `SysObj`

Nesta secção descreve-se o *use-case* que faz a sincronização das matrizes da implementação gráfica para as matrizes de sistema do `SysObj`, por intermédio de DFDs e de diagramas de sequência.

4.3.1.1. *Use-case* de Sincronização

Diagrama de Fluxo de dados (Figura 4.3.1) :

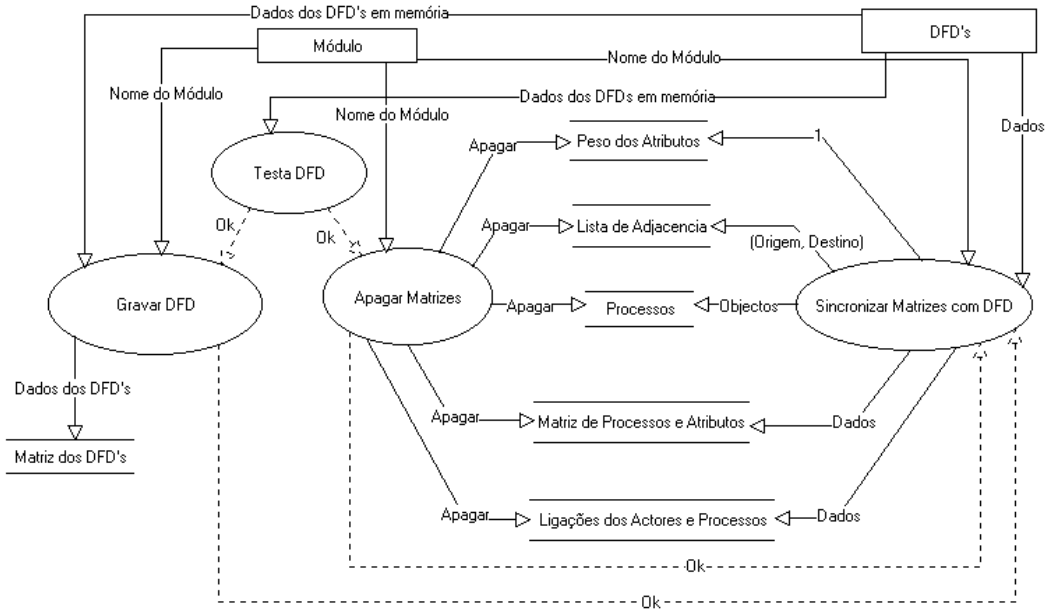

Figura 4.3.1 – *Use-case* de sincronização

Descrição do *use-case* :

- o *use-case* é iniciado com a introdução do nome do módulo actual;

- em seguida é iniciado o processo "Testa DFD" e o processo de sincronização só continua se este for bem sucedido, isto é, se o DFD estiver coerente;

- se o DFD está coerente é então iniciado o processo "Gravar DFD" com o nome do módulo anterior. Este processo grava o desenho do DFD na estrutura de dados apresentada anteriormente (actualiza a estrutura de dados "Matriz dos DFDs"). O processo de sincronização só continua se a gravação for bem sucedida, evitando-se assim, a existência de estruturas redundantes, isto é, as estruturas de dados têm que estar sempre sincronizadas;

- é então iniciado o processo "Apagar Matrizes" se os processos de teste de coerência e de gravação foram bem sucedidos. Este processo apaga todas as matrizes do `SysObj` relativas ao módulo indicado para que estas não fiquem com dados errados (lixo);

- em seguida, se os dois processos anteriores correrem bem é iniciado o processo "Sincronizar Matrizes com DFD", o que, com base nos DFDs guardados na memória e no nome do módulo, actualiza todas as tabelas da matriz do `SysObj`.

Diagramas de Sequência (Figura 4.3.2):

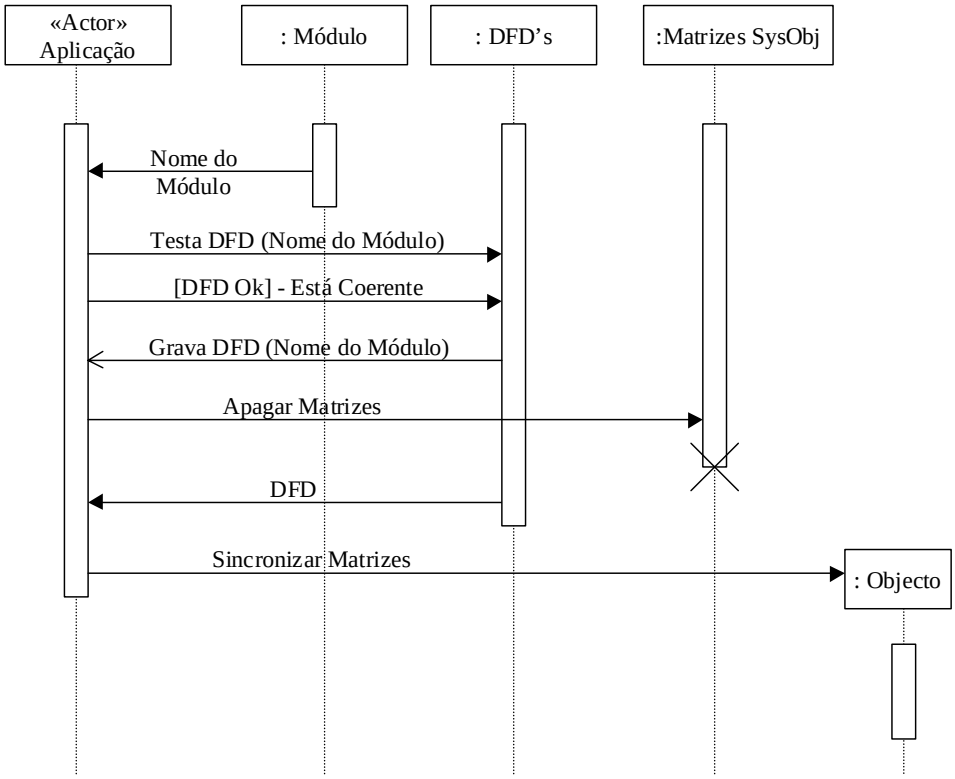

Figura 4.3.2 – Diagrama de sequência, *use-case* de sincronização

4.3.1.2. Processo "Sincronizar Matrizes com DFD"

Diagrama de fluxo de dados (Figura 4.3.3):

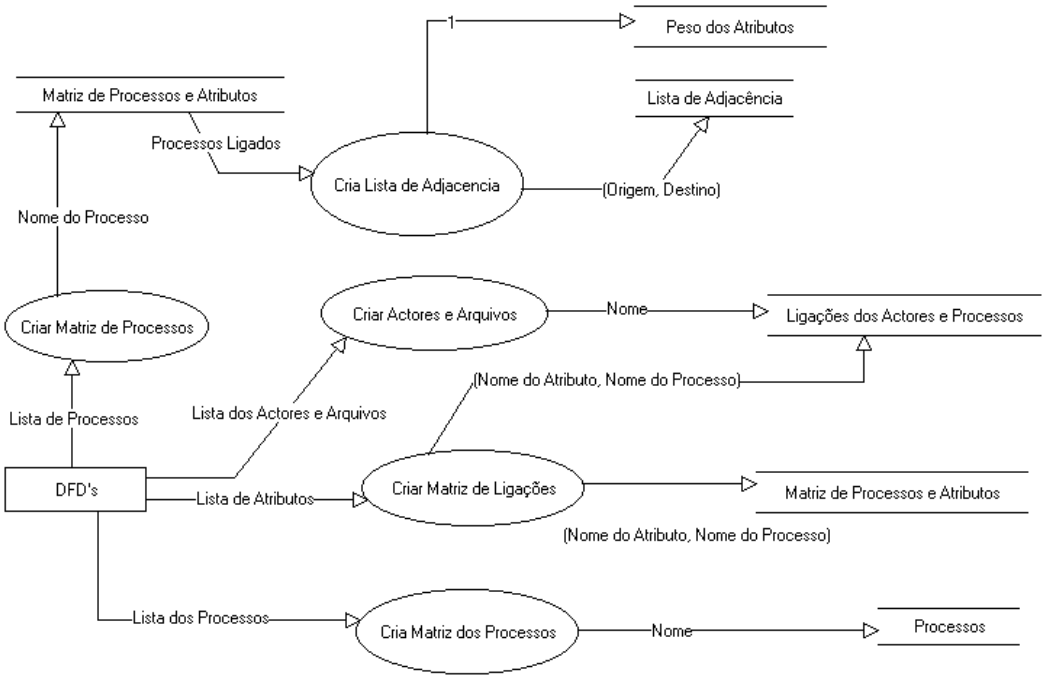

Figura 4.3.3 – Sincronizar matrizes com DFD

Descrição do processo :

- Este processo é realizado com base nos DFDs em memória;

- primeiro o processo “Cria Actores e Arquivos” gera na tabela “Ligações dos Actores e Processos” as entradas com os nomes dos actores e arquivos que estão no DFD, tendo em conta que os nomes dos arquivos são precedidos de '_r' os outros campos desses registos (que correspondem ás ligações aos processos) são preenchidos com "--------" o que significa que ainda não existem ligações;

- em seguida o processo “Cria Matriz dos Processos” cria a tabela com os processos existentes no DFD sem repetições, isto é, um processo que tem duas implementações só irá ser inscrito na tabela de processos uma única vez;

- o processo “Criar Matriz de processos” cria a “Matriz de Processos e Atributos”, tendo como colunas os processos que foram introduzidos na tabela anterior mas com o nome precedido de ‘O_’;

- posteriormente o processo “Criar Matriz de Ligações” gera os registos na “Matriz de Processos e Atributos”. Os registos correspondem a atributos, e a ligação ao processo é representada com um número na coluna correspondente ao processo que vai estar ligado por este atributo;

- a criação de entradas nas tabelas referidas está sujeita ás seguintes regras :

- se o processo em questão é processo de origem do atributo, isto é, se sai do processo para outro objecto qualquer esse campo é preenchido a '-1'. Senão o processo é processo de destino do atributo e é preenchido a '+1';

- se o atributo faz a ligação de um processo para outra implementação do mesmo processo o atributo é apenas preenchido a '1';

- o nome do registo, ou seja o nome do atributo, é atribuído com base no nome do atributo, tipo de atributo e objecto que faz a ligação ao processo;

- a criação do nome final é feito por esta ordem :
  - ✓ se o atributo de ligação for um arquivo o nome é precedido de 'R_';
  - ✓ se o atributo for de Dados é precedido de 'D_';
  - ✓ se o atributo for de Controlo é precedido de 'C_';

  - ✓ se a origem do atributo é um objecto que não um processo leva um '#' em vez do primeiro caracter '_';

  - ✓ se o destino do atributo é um objecto que não um processo leva um '*' em vez do primeiro caracter '_'.

- se o atributo não for um atributo 'processo -> processo', o processo continua e vai actualizar a matriz "Ligações dos Actores e Processos" criando um registo com o nome do actor (precedido de 'r_' no caso de ser um arquivo) de origem ou destino se este ainda não existir. Depois na primeira coluna que estiver disponível para esse actor (a coluna que tiver '---------') é introduzido o nome do atributo que faz a ligação ao processo, nome esse que já foi alterado segundo as normas supra indicadas;

- no caso de o atributo a inserir nesse actor (ou arquivo) já existir o processo é o mesmo só que o nome do atributo passa a ter o caracter '_' no lugar do caracter '#' ou '*'.

- finalmente é criada uma matriz "Lista de Adjacência" e uma outra "Peso dos Atributos" através do processo "Cria Lista de Adjacência". Dado que este processo já existia na versão anterior do `SysObj` não será descrito nesta versão.

4.3.2. Desenho de Objectos na Interface Gráfica

Nesta secção é feita a descrição do *use-case* responsável pela gestão do interface com o utilizador, a nível da introdução de *inputs* que criam desenhos assim como o respectivo desenho dos mesmos objectos no ecrã.

4.3.2.1. *Use-case* de Desenho de um Actor, Processo, Arquivo ou Texto

Diagrama de Fluxo de Dados (Figura 4.3.4) :

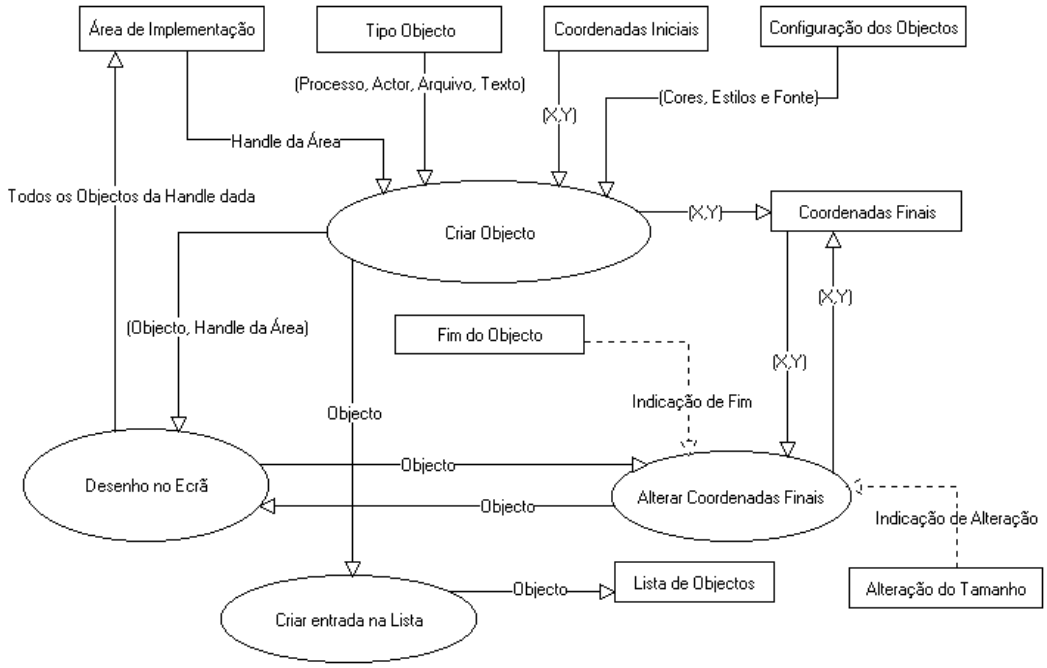

Figura 4.3.4 – *Use-case* do desenho de actores, processos, arquivos e textos

Descrição do *use-case* :

- o *use-case* de desenho de um objecto dos tipos indicados anteriormente começa com a introdução dos seguintes dados :
  - O tipo de objecto a ser desenhado;
  - A configuração visual do objecto.

- estes dados estão residentes em memória e podem ser alterados através das opções na barra de ferramentas:
  - A área de implementação onde está a ser desenhado;
  - As coordenadas iniciais do objecto.

- estes dados são passados ao *use-case* quando é feito o clique do rato ou "Mouse Down";

- a partir deste ponto existem o objecto a desenhar e a *handle* do ecrã onde ele irá ser desenhado no processo de "Desenho no Ecrã". Assim, o objecto é logo desenhado, porém é desenhado a tracejado porque ainda não foi finalizado.

Em simultâneo é criada uma entrada na lista de objectos através do processo "Cria entrada na Lista".

- o processo "Desenho no Ecrã" gera duas saídas, o desenho no ecrã físico e as novas coordenadas finais que o objecto vai tendo. Assim, enquanto o utilizador arrasta o objecto no ecrã (mantendo o botão do rato premido) as coordenadas finais são alteradas pelo actor "Alteração do Tamanho" através do processo alterar coordenadas finais. Este processo depois vai accionar o processo "Desenho no Ecrã" com o objecto passado e com as novas coordenadas a utilizar;

- quando o utilizador solta o botão do rato o objecto é finalizado pelo actor "Fim de Objecto", as coordenadas finais são actualizadas e o objecto é novamente desenhado no ecrã.

Diagrama de Sequência (Figura 4.3.5):

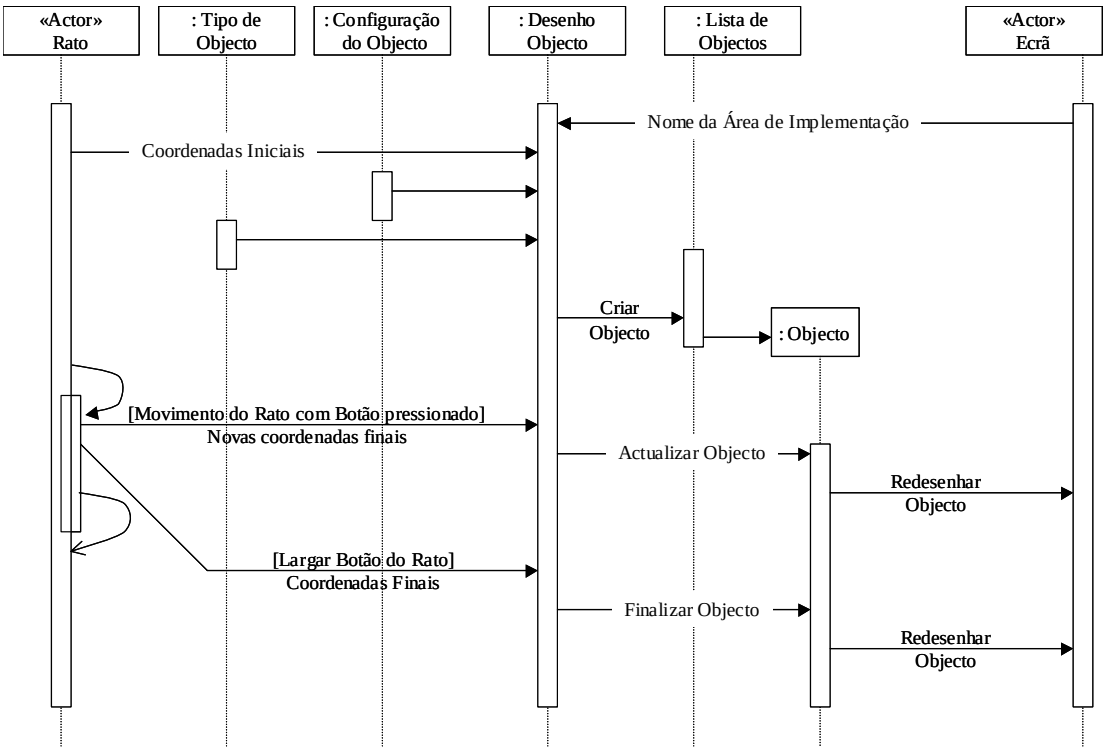

Figura 4.3.5 – Diagrama de sequência do *use-case* desenho de objectos

4.3.2.2. *Use-case* do Desenho de um Atributo

Diagrama de Fluxo de Dados (Figura 4.3.6) :

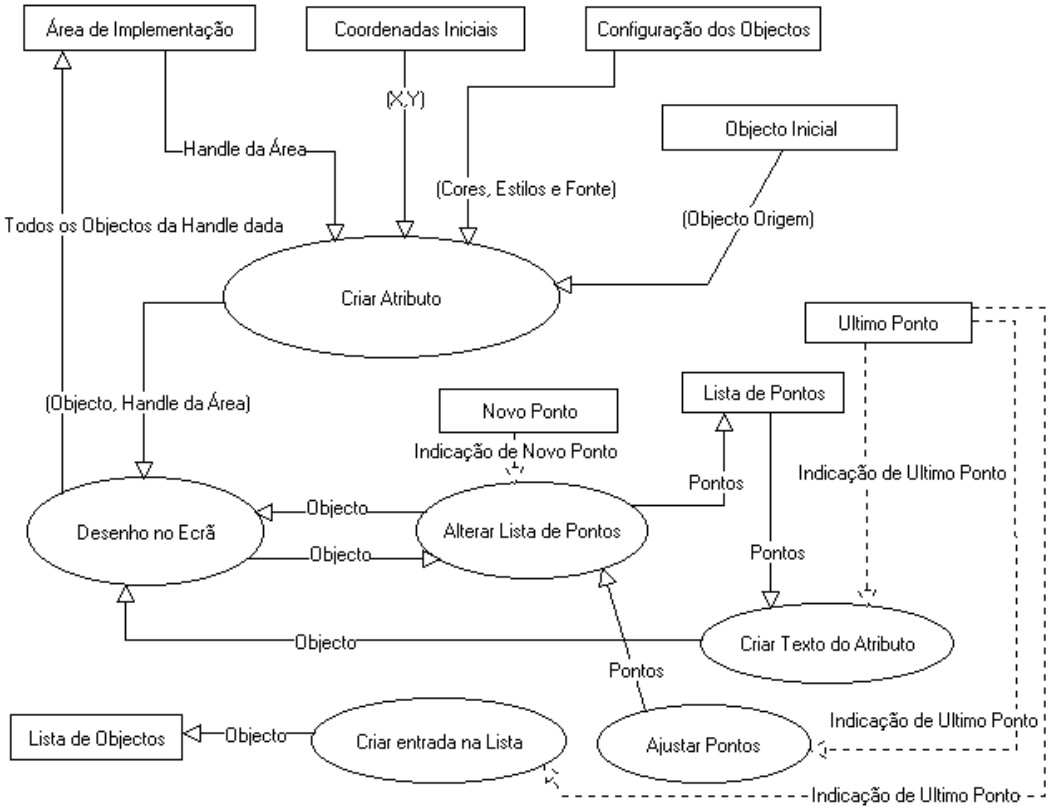

Figura 4.3.6 – *Use-case* do desenho de um atributo

Descrição do *use-case* :

- o *use-case* de desenho de um atributo começa com a introdução da configuração visual que o objecto vai ter;

- estes dados estão residentes em memória e podem ser alterados através das opções na barra de ferramentas :

    1. a área de implementação onde está a ser desenhado;
    2. as coordenadas iniciais do objecto;
    3. o objecto inicial, ou seja o objecto de origem do atributo.

- os dados são passados ao *use-case* quando o utilizador faz o clique do rato ou "Mouse Down";

- a partir deste ponto existem o objecto a desenhar e a *handle* do ecrã onde ele irá ser desenhado no processo de "Desenho no Ecrã". Assim o objecto é logo

desenhado a tracejado visto que ainda não foi finalizado. É também criada uma entrada na lista de objectos através do processo “Cria entrada na Lista”;

- o processo de “Desenho no Ecrã” gera duas saídas, o desenho no ecrã físico e as novas coordenadas finais que o novo ponto a ser criado vai tendo. Assim enquanto o utilizador arrasta a linha no ecrã as coordenadas finais são alteradas pelo actor “Novo Ponto” através do processo “alterar lista de pontos”. Este processo depois vai accionar o processo “Desenho no Ecrã” com a lista de pontos passada;

- ao fazer *click* numa área livre da área de implementação é criado pelo actor “Novo Ponto” mais um ponto no atributo. Pode existir assim uma lista infindável de pontos criados até chegar ao ponto de destino do objecto;

- quando o utilizador faz um *click* sobre um objecto esse objecto é considerado e o processo de criação do atributo termina. Assim o actor “Ultimo Ponto” dá a indicação aos processos “Criar Texto do Atributo”, “Ajustar pontos” e “Criar Entrada na Lista” de que foi feito o último ponto;

- a posição inicial do texto do atributo é calculada com base no actor “Lista de Pontos” no processo “Criar Texto do Atributo”. Este processo só é activado com a indicação de último ponto uma vez que necessita das coordenadas de todos os pontos para poder fazer uma média (horizontal e vertical). O nome do atributo também só pode ser criado no fim uma vez que o mesmo engloba o nome do objecto de origem e o nome do objecto de destino, sendo assim necessário ter o nome dos dois para concluir o nome;

- o processo “Criar entrada na Lista” cria uma entrada na lista de objectos para o atributo criado. Este processo também só pode ser realizado no fim uma vez que só no fim é que se sabe o nome do atributo;

- o processo “Ajustar Pontos” ajusta os pontos de origem e destino para que estes não fiquem dentro do objecto, isto é, os pontos (inicial e final) são puxados até á fronteira do objecto. Em seguida o objecto é redesenhado no ecrã, através do processo “Desenho no Ecrã”, já com os pontos ajustados e com o nome criado.

Diagrama de Sequência (Figura 4.3.7) :

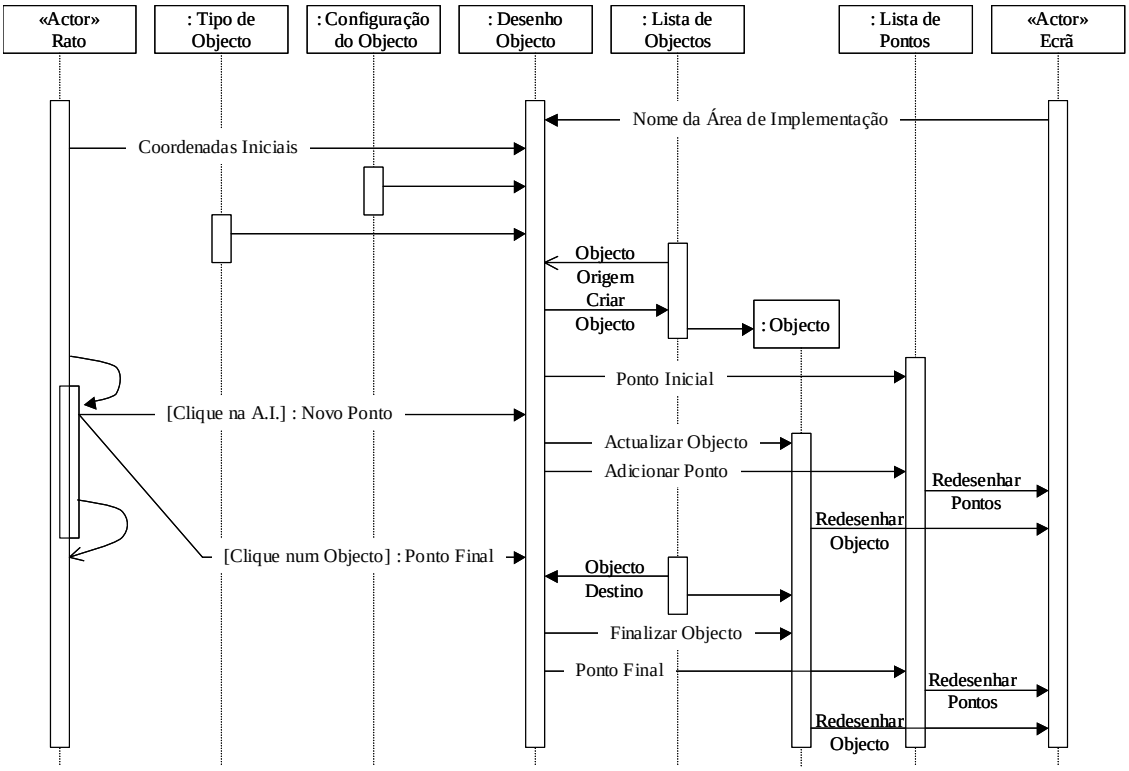

Figura 4.3.7 – Diagrama de sequência do *use-case* desenho de um atributo

4.4. Demonstração de Resultados / Exemplo de Utilização: Sistema FILL

Neste capítulo apresenta-se um caso de estudo para exemplificar a aplicação da metodologia proposta, na geração e selecção da arquitectura-alvo de objectos para um sistema *hw*/*sw*. Deste modo, é coberto um dos aspectos fundamentais da metodologia, relativos à geração, selecção e reconfiguração de arquitecturas com base na informação do plano de desempenho (um dado DFD) para selecção de uma arquitectura.

O exemplo consiste num sistema de controlo para uma máquina integrada numa linha de enchimento de garrafas de uma empresa de vinhos [RDTT98].

Vão assim ser descritos os processos de desenho dos diagramas de fluxo de dados e a interpretação dos resultados desse desenho. Posteriormente vão ser demonstrados os resultados das partições e das análises efectuadas ao caso.

4.4.1. Análise do Sistema

O caso de estudo visa a automatização de uma máquina rotuladora, (M25, M34 e M110 da figura 4.1) que se encontra integrada numa linha de enchimento de garrafas, linha 3 (M34) da figura 4.1, de uma empresa do sector vinícola, a Caves Arcos dos Reis (CAR), localizada na região Centro (Anadia) [R97a]. O sistema de automatização das linhas de enchimento de garrafas, denominado Sistema *fill*, integra um projecto mais abrangente, designado por *IntelCaves* (Caves Inteligentes), que se encontra especificado em [R97a]. Constitui objectivo genérico do Sistema *Fill* garantir que: (1) o conjunto de máquinas integradas numa linha funcionem de modo independente e interligado, e que (2) as diversas máquinas possam ser controladas e sincronizadas através de um computador de linha. Para satisfazer estes requisitos, as máquinas existentes na linha devem ser equipadas com meios que assegurem a sincronização dos equipamentos e a programação do débito da linha.

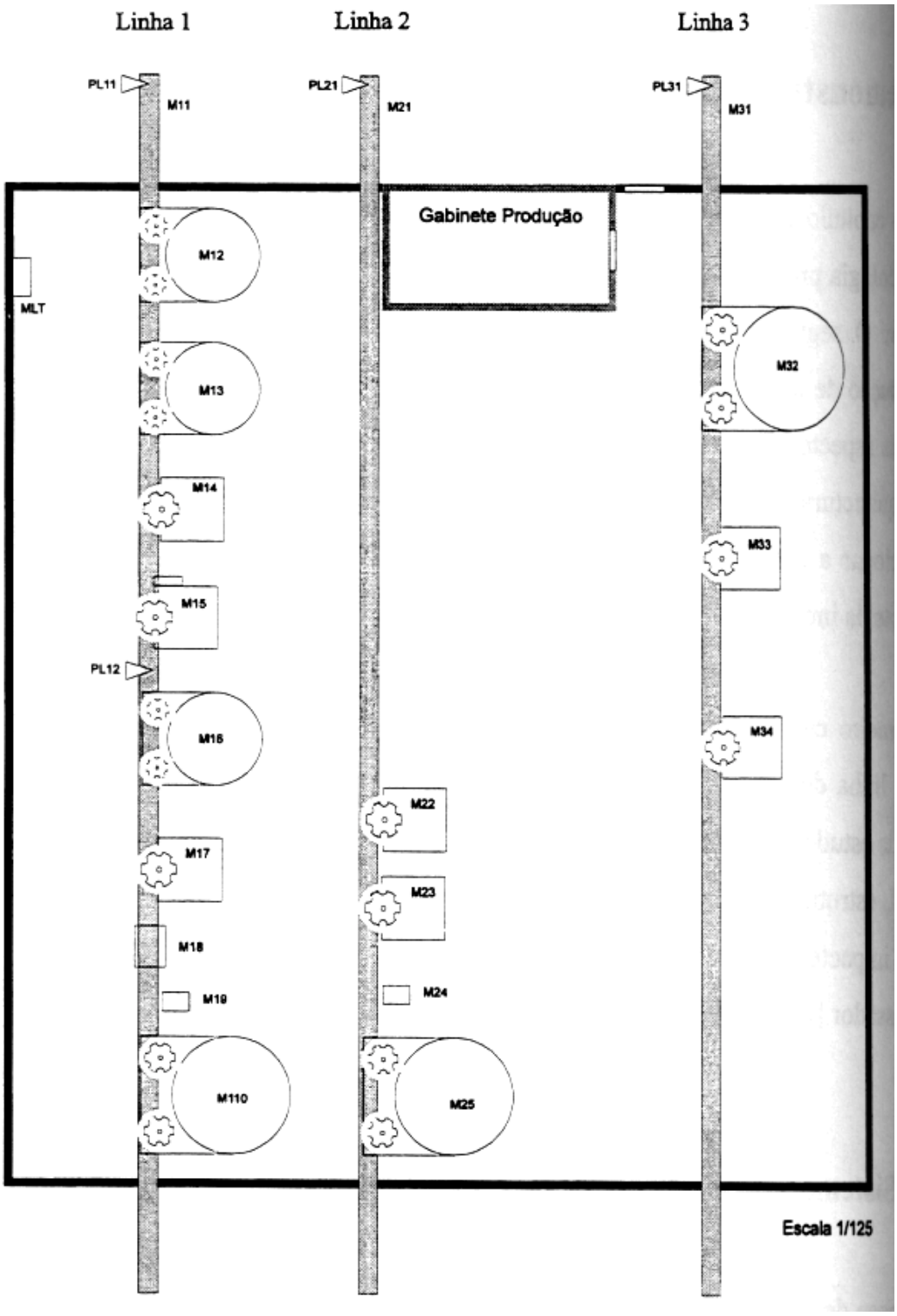

Figura 4.4.1 – Implantação das linhas de enchimento para vinhos e espumantes [R97a]

As máquinas, representadas no diagrama de implantação da figura 4.4.1, dispõem de um quadro geral, onde se concentram todas as ligações eléctricas, designadamente contactores, corte geral, protecções térmicas e circuito de segurança. Refere-se este aspecto, dado que só no quadro geral é possível ligar a máquina rotuladora e mudar o seu modo de funcionamento.

Quando as dimensões da máquina o exigem, existem botoneiras acopladas que permitem comandar o seu arranque e paragem, através de botões com encravamento.

A máquina rotuladora é, do ponto de vista mecânico, a máquina mais complexa de uma linha de enchimento. Faz uso de um tambor de estrelas de entrada e de saída e possui várias peças de substituição, nomeadamente os cilindros engomadores, extractores e distribuidores de cola, bem como os carregadores de rótulos, dependentes da campanha de produção. Para a linha 3, as restantes máquinas são a enchedora (M32 da figura 4.4.1) e a rolhadora (M33 da figura 4.1).

O sistema *Fill* (Enchimento) é um sistema modular baseado em macro-componentes de *hw* comerciais que permitem flexibilizar a sua evolução e personalização a diferentes linhas de enchimento bem como a outras unidades industriais. Assim, embora se vise como objectivo central de um sistema **integrado** de controlo da linha, o sistema *fill* pretende assegurar uma arquitectura de **controlo distribuído**, que descentralize a capacidade de intervenção, personalizando o controlo de cada máquina através da programação de uma unidade de processamento local. Na figura 4.4.2 representa-se a arquitectura modular de alto nível do sistema.

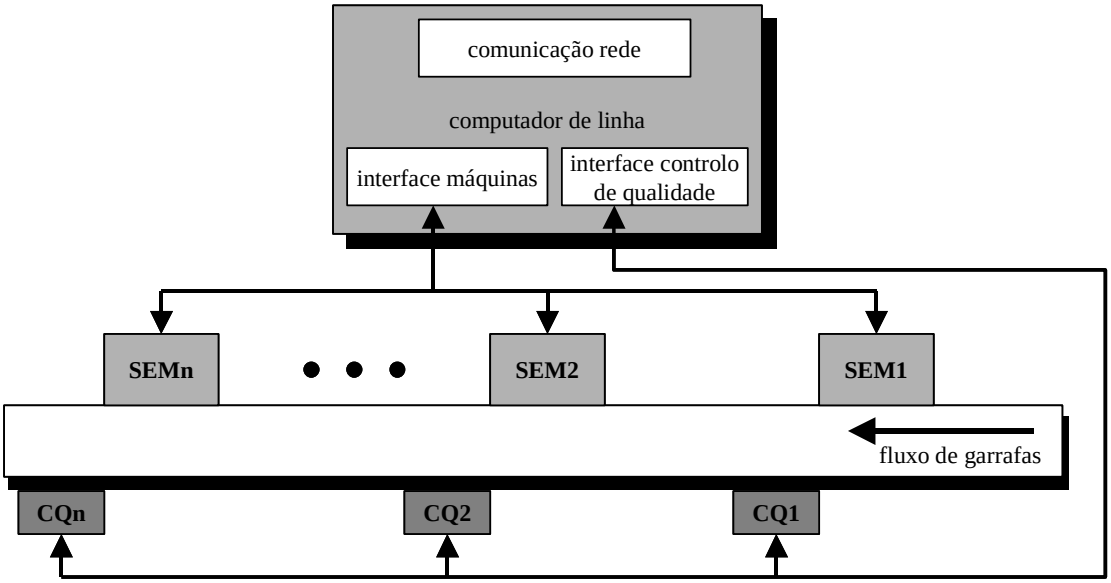

Figura 4.4.2 – Diagrama de alto nível dos módulos do sistema *fill* [R97a]

O computador de Linha assegura a comunicação com o sistema *IntelCaves* através do módulo "Comunicação de Rede". O controlo e gestão das linhas (exemplificado para uma) é feito através das interfaces "Máquinas" e "Controlo de Qualidade". A cada máquina associa-se um SEM (Sistema Electrónico da Máquina) e um módulo C.Q. (Controlo de Qualidade).

Na figura 4.4.3 apresentam-se os blocos pré-definidos para o sistema electrónico de cada módulo associado ao controlo de uma máquina da Linha, a fim de garantir a modularidade pretendida. Os blocos sombreados existiam antes da automatização da Linha. Os módulos restantes destinam-se à automatização da Linha.

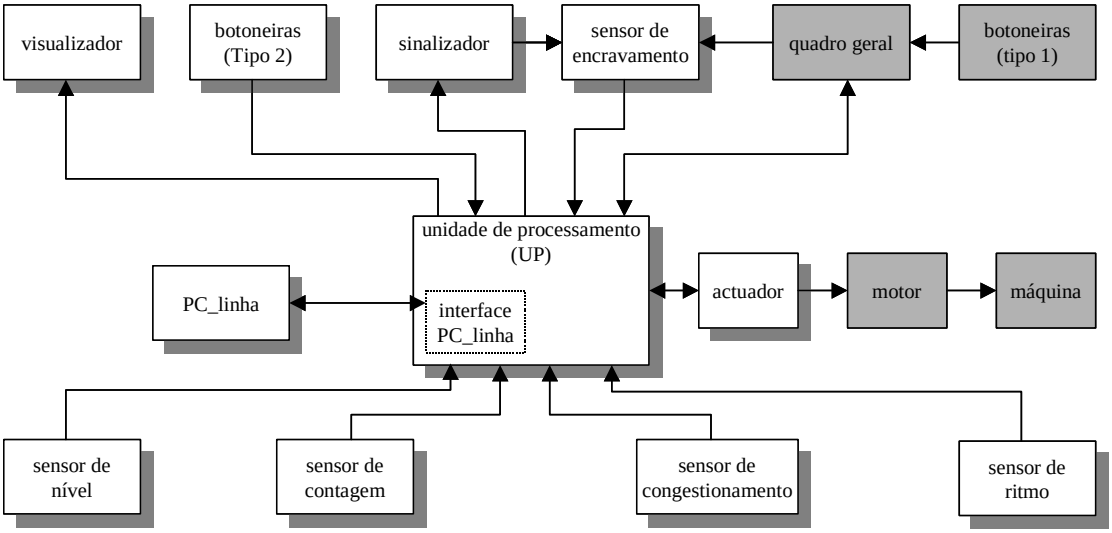

Figura 4.4.3 – Sub-blocos do sistema de controlo (SEMi) para cada máquina [R97a]

A **Unidade de Processamento** (UP) é responsável pelo controlo da máquina. O **Visor** mostra a informação básica sobre a situação em que a máquina se encontra. O **Actuador** controla o motor da máquina, evitando ainda o uso de contadores e de módulos inversores neles baseados. As **Botoneiras** são caixas com botões de comando, que permitem realizar algumas operações sobre a máquina respectiva, nomeadamente paragem de emergência (tipo 1), e paragem normal, marcha-lenta-à-frente, marcha-lenta-atrás e carga de rótulos (tipo 2). A configuração destas botoneiras dependerá do tipo de máquina em que são instaladas. Finalmente, a **Interface com o Computador de Linha** (embutida na UP) tem por objectivo garantir toda a comunicação entre o sistema local SEM e o computador de linha.

A fim de proceder à monitorização e controlo da Linha de produção utilizam-se sensores para detecção: (1) do ritmo de processamento das garrafas, (2) do nível de garrafas acumuladas na entrada da máquina, (3) do número de garrafas processadas, (4) da existência de congestionamento de garrafas na saída da máquina e (5) da situação de encravamento da máquina. A disposição relativa dos sensores está representada na figura 4.4.4, onde: **1** é o sensor de ritmo, **2** é o sensor de nível, **3** é o sensor de contagem, **4** é o sensor de encravamento e **5** é o sensor de congestionamento.

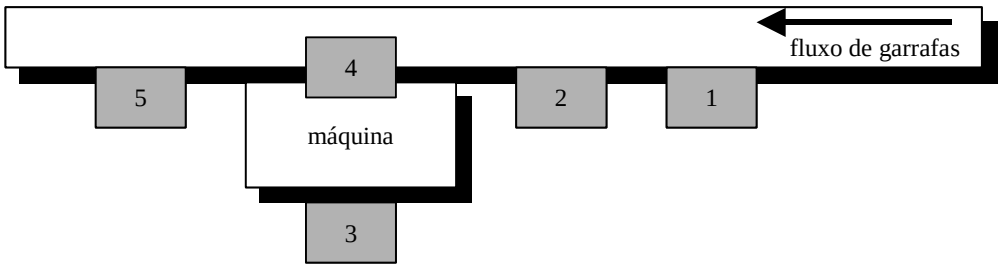

Figura 4.4.4 – Disposição dos sensores [R97a]

O **Sinalizador** destina-se a produzir um sinal sonoro e visual, quando varia a velocidade da máquina respectiva, e o **Motor** garante o movimento da máquina.

O caso de estudo refere-se ao projecto da funcionalidade da Unidade de Processamento (UP) do SEM da máquina rotuladora. Esta funcionalidade será implementada em *sw*, e distribuída por objectos que serão atribuídos ao nó "unidade de processamento" representado na figura 4.4.3.

O facto de se ter optado pelo projecto da UP do SEM da máquina rotuladora não retira generalidade ao caso de estudo, no contexto das linhas de enchimento de garrafas, uma vez que os módulos SEM são semelhantes a todas as máquinas da linha. A fim de definir os *use-case* responsáveis pela funcionalidade do sistema torna-se necessário identificar os estados em que o sistema deve funcionar, o que provoca as transições entre eles e as operações a executar em cada estado. De facto, tratando-se de um sistema de tempo real, a componente dinâmica condiciona todo o funcionamento e deve constituir o ponto de partida da análise.

Assim, considera-se que a linha de enchimento deve poder encontrar-se num dos seguintes seis estados possíveis:

- Produção
- Parada
- Configuração
- Manutenção
- Avaria
- Carga de Rótulos

aos quais corresponde o diagrama de estados da figura 4.4.5. Toda a informação referente ao funcionamento da linha é descrita em detalhe em [R97a]. A funcionalidade associada a cada *use-case*, que passa a descrever-se, foi deduzida desse documento. Identificaram-se 9 *use-cases*.

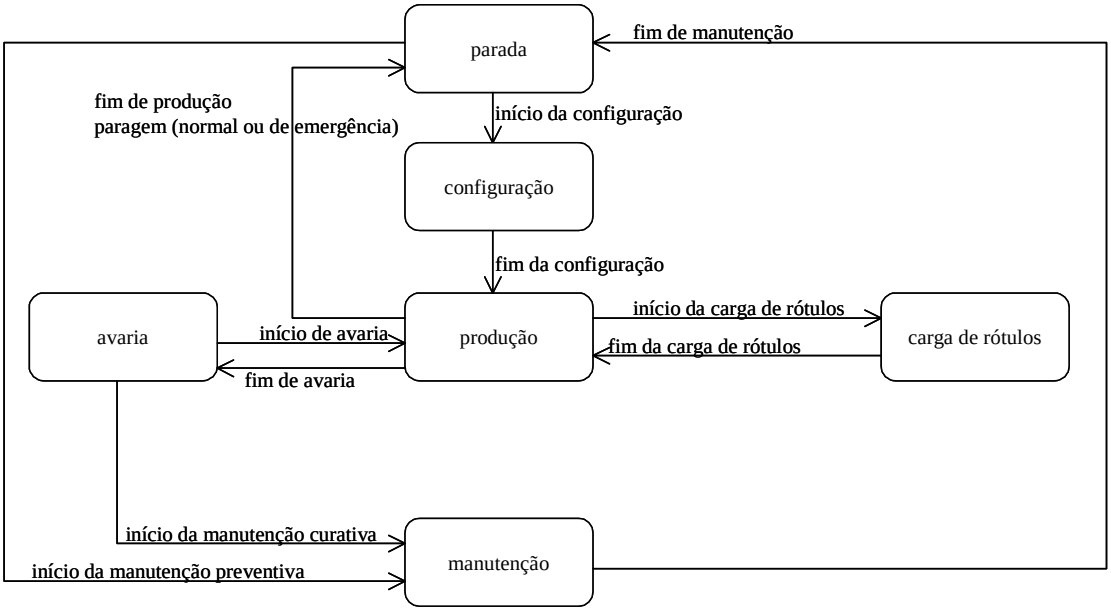

Figura 4.4.5 – Diagrama de estados da linha [R97a]

4.4.2. Exemplo de Utilização

Para iniciar o projecto no `SysObj` é necessário criar um projecto que no exemplo apresentado tem o nome *fill*. Para isso o utilizador deve aceder ao menu *"File|New Project"* (Figura 4.4.6)

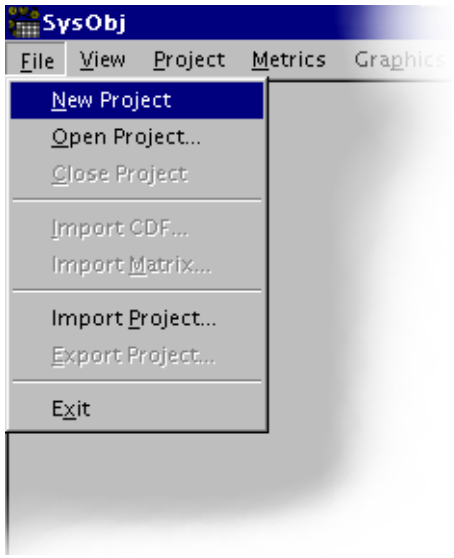

Figura 4.4.6 - Menu de criação de projecto

Em seguida é atribuído o nome - *fill* - ao projecto que será desenvolvido apenas com um módulo (Figura 4.4.7).

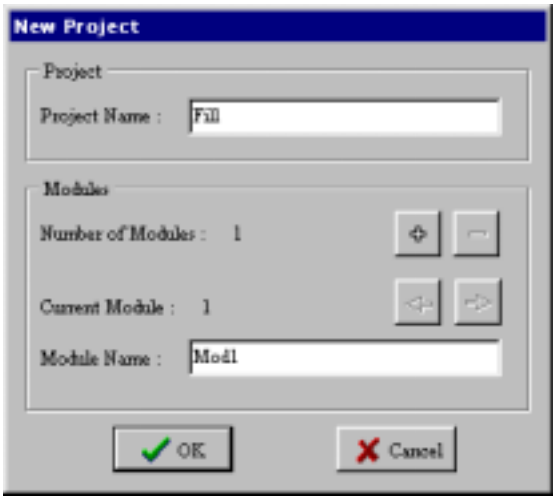

Figura 4.4.7 – Criação do projecto

Para iniciar o desenho dos diagramas de fluxo de dados, no módulo acabado de criar, selecciona-se *"Project|DFD Graphic Interface"* (Figura 4.4.8).

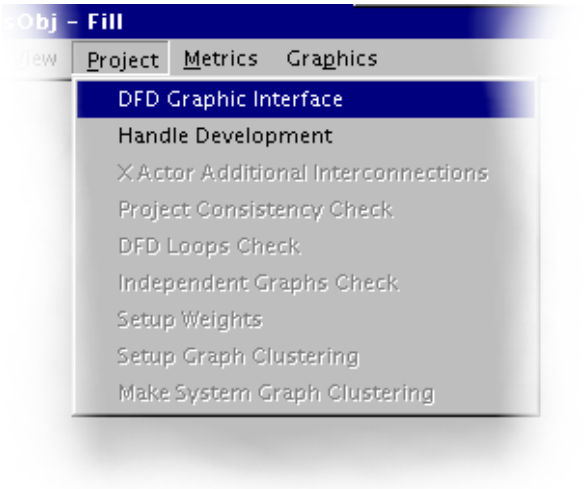

Figura 4.4.8 – Menu da interface gráfica

Ao entrar no ecrã da interface gráfica é necessário criar uma área de implementação para cada *use-case*, para tal basta clicar no botão da barra de ferramentas do projecto, ilustrado na figura 4.4.9.

Figura 4.9 – Botão de criação de uma área de implementação

É sempre possível alterar o nome do *use-case*, ou área de implementação, para o nome do *use-case* que vai ser desenhado isto porque ao criar uma área de implementação nova ela é criada com um nome por omissão típicamente "*Implementation Area[x]*", onde x é o número de áreas de implementação existentes. Para isso clicar na área de implementação com o botão do lado direito do rato e é apresentado ao utilizador o menu de atalhos dessa área de implementação. Em seguida escolher a opção *"Implementation Area Name"* (Figura 4.4.10).

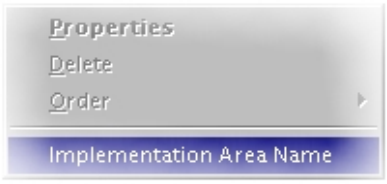

Figura 4.4.10 – Opção de mudar o nome da área de implementação

Após esta operação surge a caixa de diálogo que permite alterar o nome da área de implementação, bastando introduzir o nome pretendido seguido de um *click* em *"Apply"* (Figura 4.4.11).

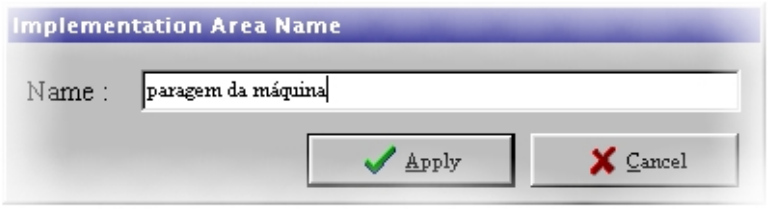

Figura 4.4.11 – Alteração do nome da área de implementação

Neste exemplo apenas vai ser utilizado o *use-case* paragem da máquina, para descrever a técnica de desenho e de utilização da interface gráfica. Quanto aos restantes 8 *use-cases* apenas serão descritos e apresentados já desenhados.

***Use-case* paragem da máquina**

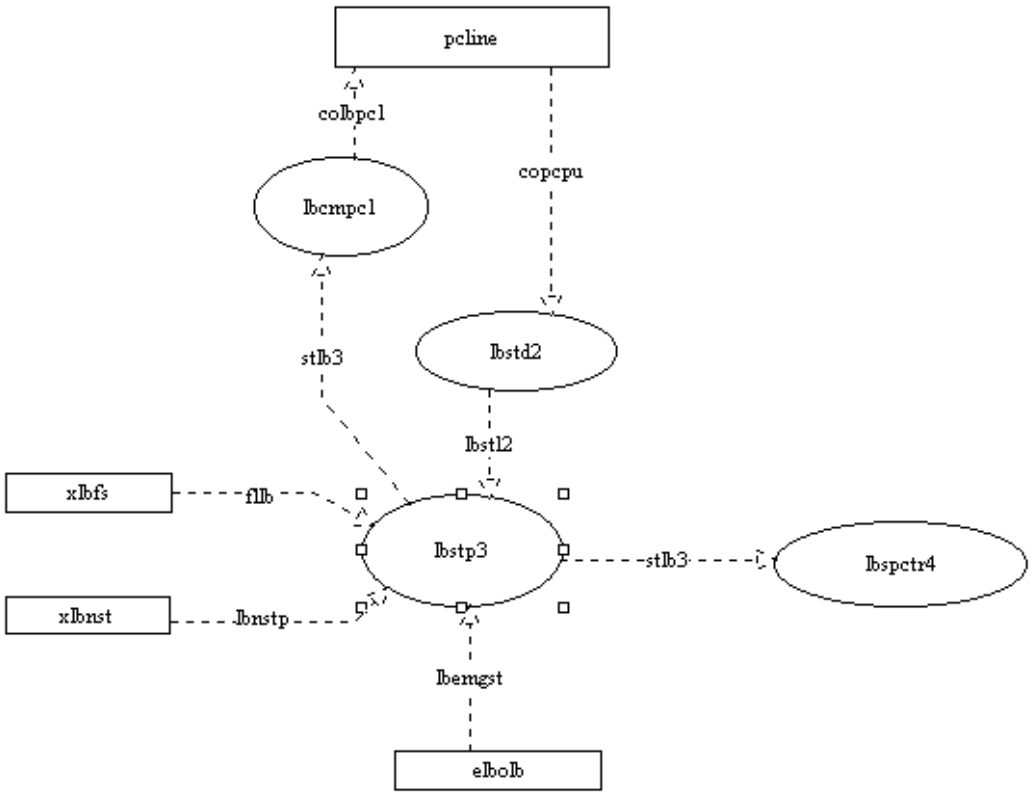

Figura 4.4.12 - Diagrama do *use-case* "paragem da máquina"

A paragem da máquina deve verificar-se quando:

- o PC da linha envia uma ordem de paragem;
- o sensor de encravamento da máquina actua;
- o botão de paragem normal da máquina é activado;
- o quadro eléctrico da máquina é desligado, por acção do botão de paragem de emergência.

O computador da linha envia uma ordem de paragem quando:

- recebe um pedido de paragem da linha feito pelo operador através do seu teclado;
- a linha entra no estado de manutenção (a pedido do computador do serviço de manutenção);
- a linha entra em estado de avaria;
- a linha entra no estado de carga de rótulos;
- sensor de encravamento de uma das máquinas é activado;
- botão de paragem normal de uma das máquinas é activado;
- quadro eléctrico de uma das máquinas é desligado por acção do botão de paragem de emergência.

Para que a máquina saia do estado "parada" é necessário que a causa tenha sido eliminada e que o operador introduza no computador da linha o relatório da causa da paragem.

Desenho do *use-case* paragem da máquina

Foi desenhado em primeiro lugar o actor *"pcline"*. Para isso foi seleccionado o botão de desenho de actores, representado na figura 4.4.13, e depois premido o botão do rato na área de implementação e arrastando-o por forma a criar um rectângulo, figura 4.4.14, soltando o botão quando o objecto tiver o tamanho desejado. Enquanto o programa estiver em modo de desenho, isto é com o botão do rato pressionado, a linha do objecto aparece a tracejado.

Figura 4.4.13 – Botão de desenho de actores

Figura 4.4.14 – Criação do actor

Depois de criado o actor, figura 4.4.15, é necessário alterar-lhe o nome uma vez que este é criado com um nome sequencial (actor1, actor2, actor3, etc...). Para isso deve em primeiro lugar ser seleccionado o objecto, então é necessário activar o botão de selecção que está representado pela figura 4.4.16.

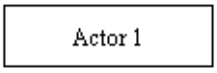

Figura 4.4.15 – Objecto criado com nome por defeito

Figura 4.4.16 – Botão de selecção de objectos

Ao fazer *click* sobre o objecto ele fica seleccionado, como pode ser verificado através das marcas de redimensionamento visíveis em volta do objecto (Figura 4.17).

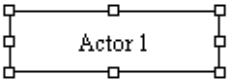

Figura 4.17 – Objecto seleccionado

Em seguida ao fazer *click* com o botão do lado direito do rato sobre o objecto é apresentado o menu de atalhos do objecto, figura 4.4.18.

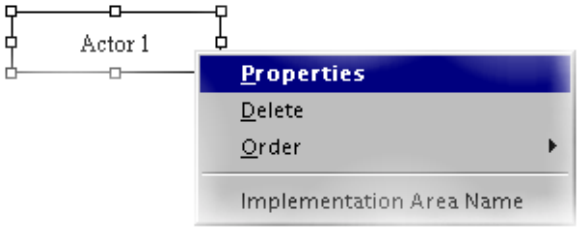

Figura 4.4.18 – Menu de atalhos do objecto

Após esta operação, e ao escolher *"Properties"* no menu de atalhos, aparece uma caixa de diálogo com as propriedades do objecto, ou seja, o nome da classe do objecto, *"class"*, o nome do objecto, *"name"*, e o tipo de objecto, que neste caso é *"Actor"*. Nesta caixa alteram-se então os dois nomes para *pcline*, uma vez que se pretende criar uma implementação da classe *pcline* com o nome *pcline.* Esta acção é terminada com a escolha de *"Apply"*, figura 4.4.19.

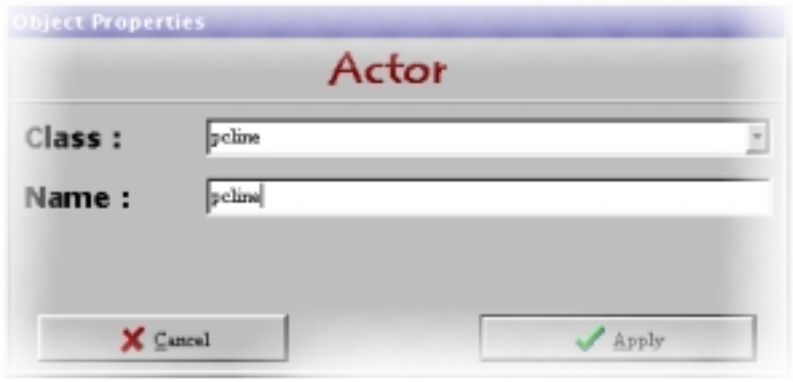

Figura 4.4.19 – Propriedades do objecto

O próximo passo será desenhadar o processo *"lbcmpc1"* que faz a comunicação entre a unidade de processamento e o pc da linha *"pcline"*. O desenho deste objecto e a modificação do seu nome, é feita de forma idêntica á utilizada para desenhar o actor *"pcline"*, porém, para iniciar o desenho deve seleccionar-se o botão de desenho de processos (Figura 4.4.20).

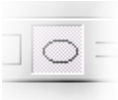

Figura 4.4.20 – Botão de desenho de processos

Após ter sido desenhado o processo “lbcmpc1” é criado o atributo “colbpc1” que liga o actor “pcline” e o processo “lbcmpc1”. Para isso selecciona-se o botão de desenho de atributos (Figura 4.4.21).

Figura 4.4.21 – Botão de desenho de atributos

Após seleccionar o botão de desenho de atributos deve ser efectuado um *click* dentro do processo e depois *click* dentro do actor. De realçar que enquanto o atributo está a ser criado a linha que o compõe está a tracejado (Figura 4.4.22).

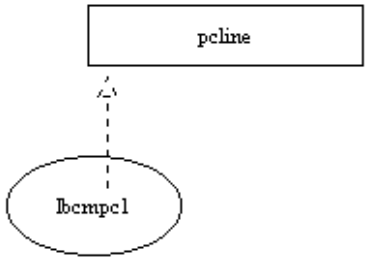

Figura 4.4.22 – Criação de um atributo

Depois de ter sido criado o atributo, este deve ser seleccionado e em seguida fazer *click* com o botão do lado direito do rato sobre o atributo para obter o seu menu de atalhos. De seguida seleccionar “Properties” a fim de lhe alterar o nome para “colbpc1”. Este procedimento de alteração do nome é idêntico para os processos e para os actores tendo já sido descrito (Figura 4.4.19).

Por fim, altera-se a o tipo da linha para tracejado para marcar o atributo como sendo um atributo de controlo. Para isso, seleccionar o atributo, fazendo *click* sobre a linha ou no texto do atributo, e utilizar o botão de definição de tipo de linha (Figura 4.4.23). Qualquer tipo de linha diferente da contínua é considerado pelo `SysObj` como um atributo de controlo.

Figura 4.4.23 – Botão de selecção do tipo de linha

Depois de alterar o tipo da linha para tracejado o resultado obtido deve ser semelhante ao descrito na figura 4.4.24, dependendo do tipo de tracejado utilizado.

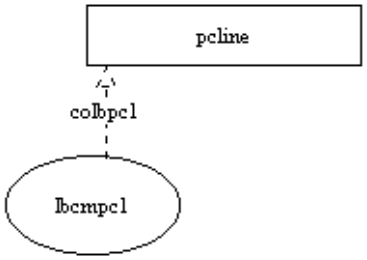

Figura 4.4.24 – Atributo de controlo desenhado

***use-case*: máquina em produção** (figura 4.4.25)

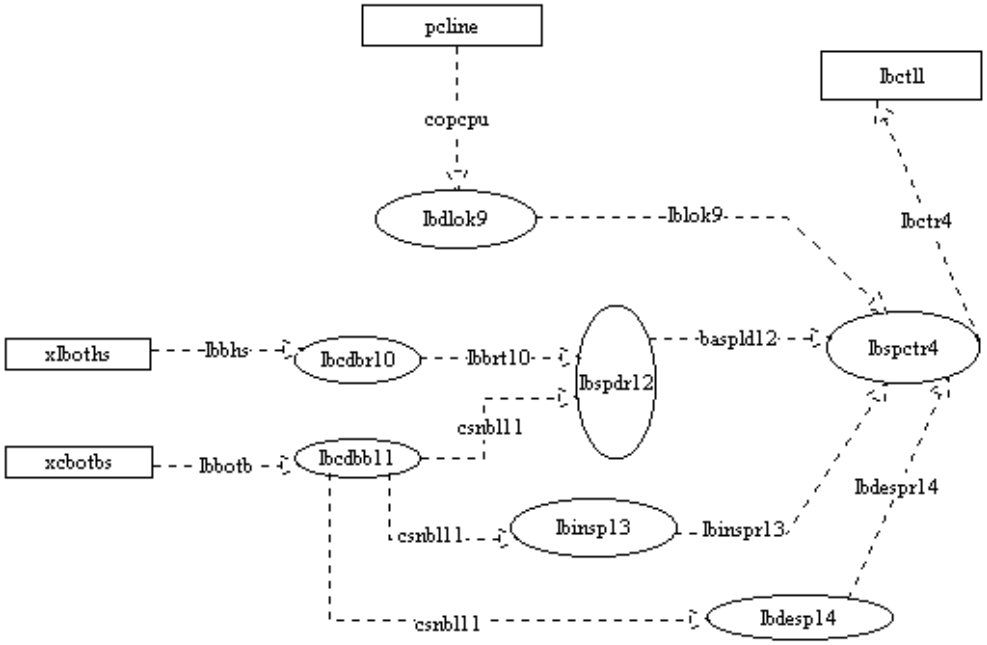

Figura 4.4.25 - Diagrama do *use-case* "máquina em produção"

A máquina pode entrar em produção se:

- receber informação do sensor de ritmo, e
- receber informação do sensor de nível, e
- receber informação do computador da linha que esta está pronta para entrar em produção.

A informação do computador da linha de que a máquina está em condições para entrar em produção consiste na activação do sinal "linha *ok*", que depende das seguintes
informações:

- a linha não está no estado "carga de rótulos", e
- a linha não está no estado "avaria", e
- a linha não está no estado "configuração", e
- nenhum botão de paragem normal está activado, e
- foi feita a verificação da ficha de produção, e
- foi introduzido o relatório da última paragem.

A velocidade da rotuladora é ligeiramente inferior à da enchedora, o que dá origem a que se acumulem garrafas na entrada da rotuladora. O ajuste de velocidade desta máquina, como em todas as máquinas escravas (*slaves*), é feito com base na informação do sensor de nível, que controla a velocidade da máquina de forma a manter na entrada a quantidade de garrafas que permita o processamento sem interrupções da produção.

***use-case*: configuração da máquina** (figura 4.4.26)

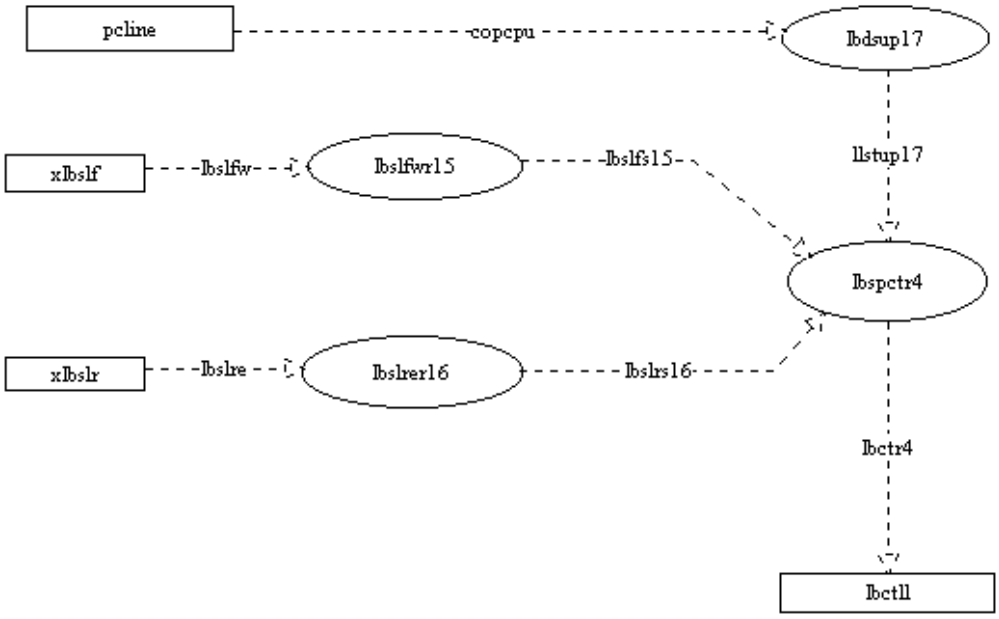

Figura 4.4.26 - Diagrama do *use-case* "configuração da máquina"

A máquina está em configuração quando:

- recebe informação do computador da linha que está no estado "configuração"

Enquanto estiver neste estado, a máquina pode ser accionada a partir do comando local, por intermédio dos botões "marcha-lenta-à-frente" e "marcha-lenta-atrás".

A máquina permanece neste estado enquanto:

- a configuração não está concluída
- não tenha sido introduzido no computador da linha o relatório da última paragem
- não tenha sido actualizada/validada a ficha de produção da campanha (tabela de configuração da linha).

***use-case*: correcção da velocidade *versus* congestionamento na saída** (figura 4.4.27)

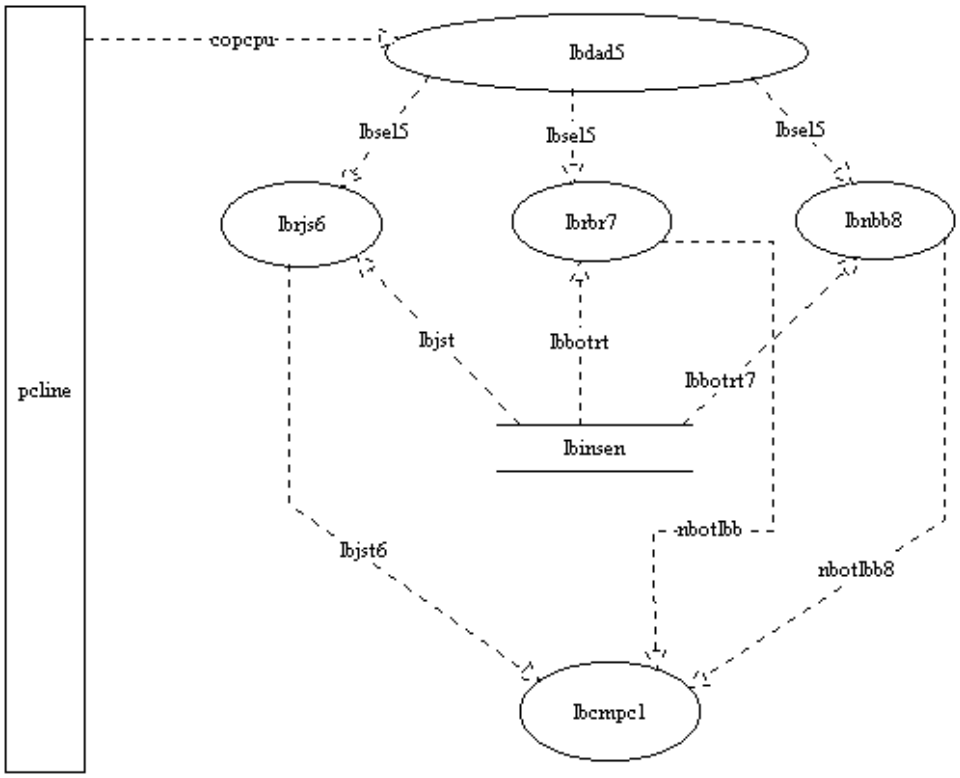

Figura 4.4.27 - Diagrama do *use-case* "correcção da velocidade *versus* congestionamento na saída"

Por intermédio da leitura do estado dos sensores de ritmo (número de garrafas/hora) e de nível (número de garrafas na entrada da máquina) o PC da linha pode determinar o incremento/hora do número de garrafas na entrada da máquina. Assim, quando o sensor de congestionamento da rotuladora é activado, o que significa que existe acumulação de garrafas na fase de embalagem (processo manual na saída da linha), o PC da linha fornece novo valor de velocidade à máquina directora (a primeira máquina da linha, M32, da figura 4.4.1). As restantes máquinas, escravas, farão o seu ajuste de velocidade em função da nova cadência de chegada de garrafas, com base nos seus sensores de ritmo e de nível.

***use-case*: carga de rótulos** (figura 4.4.28)

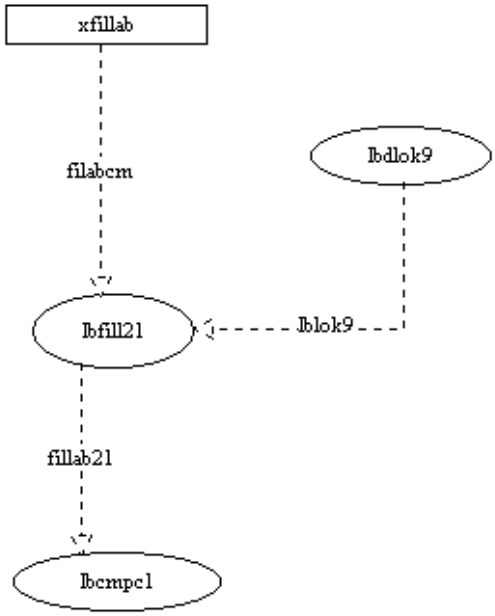

Figura 4.4.28 - Diagrama do *use-case* "carga de rótulos"

Para que a máquina entre no estado de carga de rótulos é necessário que a linha tenha feito uma paragem normal a partir do computador da linha e que o operador actue o botão da carga de rótulos. A saída deste estado verifica-se quando o operador introduz no computador da linha o relatório com a justificação da paragem.

***use-case*: actualização do estado dos sensores** (figura 4.4.29)

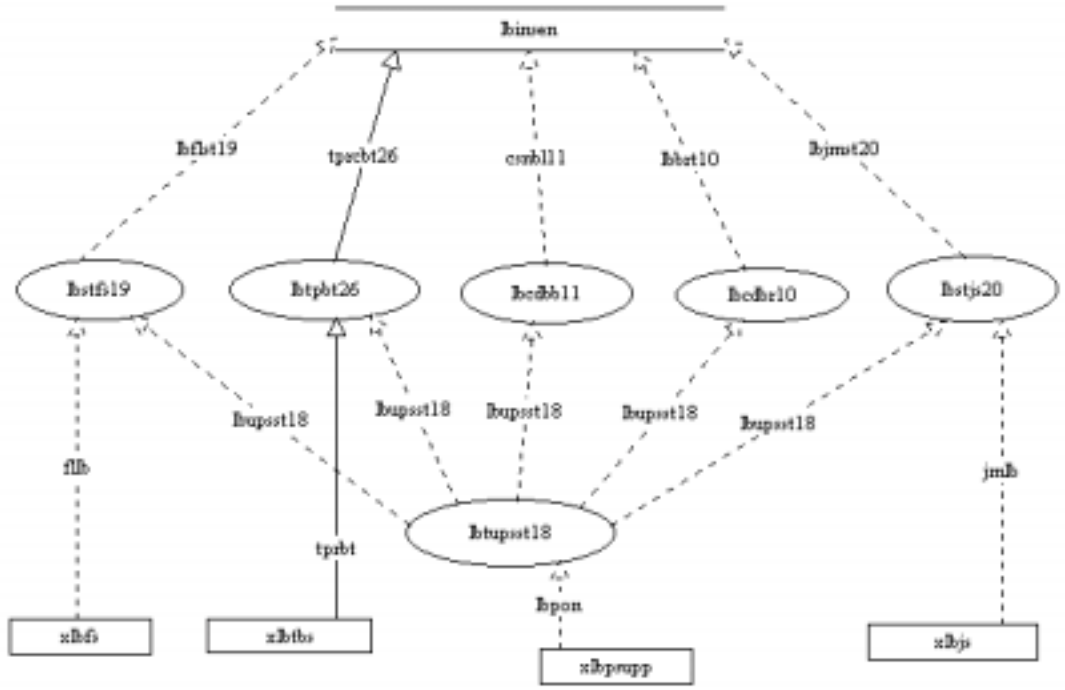

Figura 4.4.29 - Diagrama do *use-case* "actualização do estado dos sensores"

Com uma periodicidade a ajustar num nível mais baixo do desenvolvimento do sistema, os estados dos sensores da máquina são actualizados num registo, para posterior leitura pelo PC da linha, que fará um varrimento ao estado dos sensores, com intervalos de tempo a definir em estágios posteriores do projecto.

***use-case*: informação para o estado da linha** (figura 4.4.30)

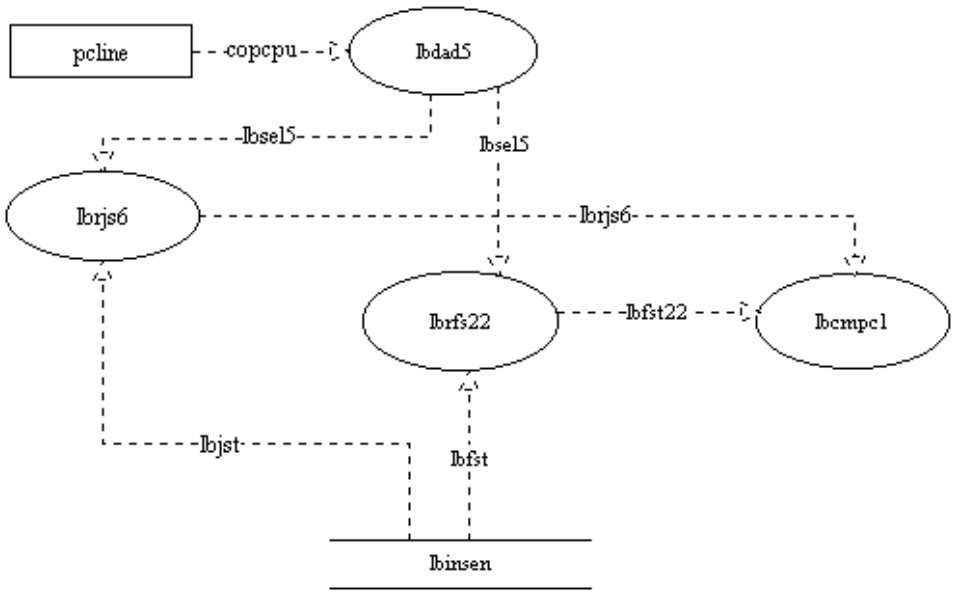

Figura 4.4.30 - Diagrama do *use-case* "informação para o estado da linha"

A leitura dos estados dos sensores de cada máquina permite gerir o controlo da linha e determinar em cada instante em que estado ela se encontra, ou para que estado deve transitar.

No caso particular da rotuladora, o computador de linha necessita de saber, para este fim, o estado dos sensores de encravamento e de congestionamento.

***use-case*: aviso de variação de velocidade** (figura 4.4.31)

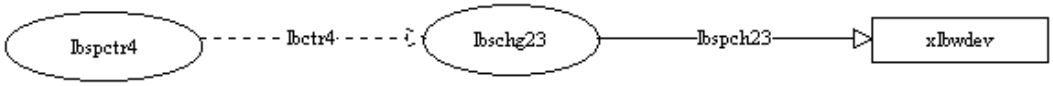

Figura 4.4.31 - Diagrama do *use-case* "aviso de variação de velocidade"

Sempre que a velocidade da máquina varia, por paragem, arranque ou auto ajuste, é gerado um sinal que estimula um sinalizador visual e acústico, para segurança dos operadores.

***use-case*: aviso da situação da máquina** (figura 4.4.32)

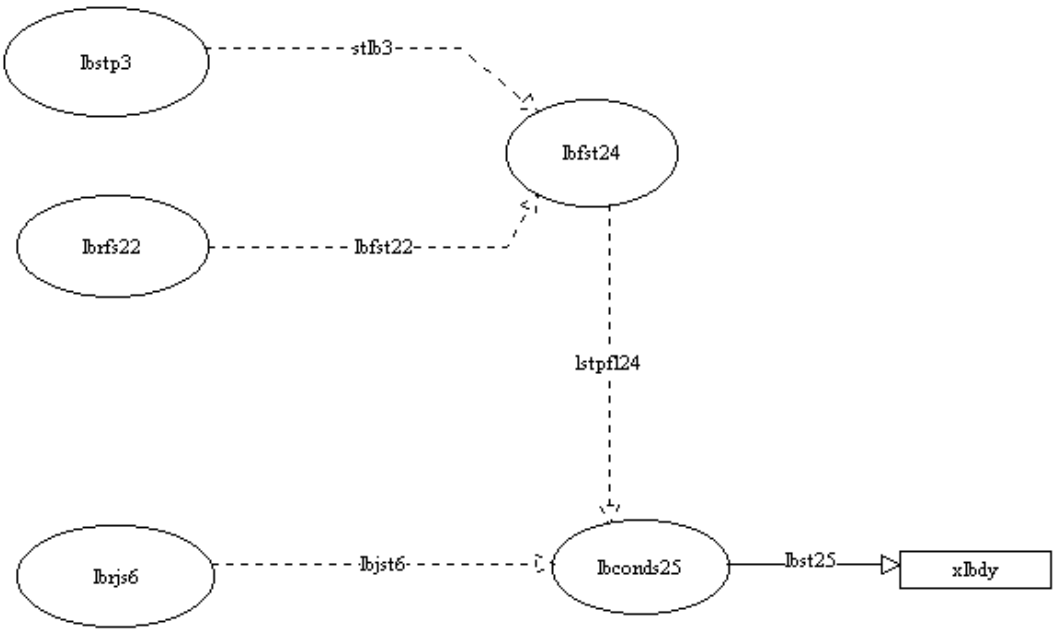

Figura 4.4.32 - Diagrama do *use-case* "aviso da situação da máquina"

Em função do estado dos sensores da máquina é gerada informação que é enviada para um visor, que informa sobre a situação da máquina. Os estados possíveis da rotuladora são:

- Em produção
- Carga de rótulos
- Parada
- Encravada
- Congestionada.

A partir da leitura e interpretação da especificação do sistema, constata-se que não foram identificados *use-cases* relativos aos estados "avaria" e "manutenção". De facto, no contexto do sistema em desenvolvimento, a máquina está no estado "avaria", quando o sensor de congestionamento da máquina *i*, está activado e o sensor de encravamento da máquina *i*+1 não está activado. Deste modo o excesso de garrafas na saída da máquina *i* (sensor de congestionamento activado) fica a dever-se a uma avaria da própria máquina *i*. Dado que a máquina em estudo (a rotuladora) é a última da linha, ela nunca estará formalmente em estado de avaria (não existe a máquina seguinte), embora a linha possa estar naquele estado devido às outras duas máquinas. Assim, o estado de avaria da rotuladora, fica incluído no estado de "máquina parada". De facto, esta máquina pode parar por exemplo devido à actuação do sensor de encravamento, o que pode ser indicador de uma avaria na máquina. Porém esse facto fará com que a linha entre no estado "parada" e não no de "avaria", sendo esta a razão porque não foi identificado um *use-case* que descrevesse o estado de avaria da rotuladora.

Relativamente ao estado de “manutenção”, ele faz sentido para a linha mas não para uma máquina, uma vez que o controlo de uma máquina ao receber uma ordem de paragem, eventualmente a pedido da manutenção, não tem necessidade de identificar a sua origem. Deste modo, o estado de manutenção da linha é a nível de máquina incluído no estado de “máquina parada”.

4.4.3 Demonstração de Resultados

Nas tabelas 4.4.2 e 4.4.3, representam-se os dicionários dos processos e atributos do DFD da UP do SEM da máquina rotuladora e dos actores que interagem com este sistema.

Funcionalidade (***use-case***)	**Modelo das funções**		
	Processos	**Origem da informação**	**Destino da informação**
paragem da máquina	p1	p3	pcline
	p2	pcline	p3
	p3	xlbfs xlbnst elbolb p2	p4 p1
	p4	p3	lbtcll
máquina em produção	p4	p9 pl2 pl3 pl4	iblcll
	p9	pcline	p4
	p10	xlboths	pl2
	pll	xebotbs	pl2 pl3 pl4
	pl2	p10	p4
	pl3	pll	p4
	pl4	pll	p4
configuração da máquina	p4	pl5 pl6 pl7	ibtcll
	pl5	xlbsif	p4
	pl6	xlbsir	p4
	pl7	pcline	p4

Tabela 4.4.1 - Modelo Funcional da UP do SEM

Funcionalidade (*use-case*)	**Modelo das funções**		
	Processos	**Origem da informação**	**Destino da informação**
correcção da velocidade *versus* congestionamento na saída	p1	p6 p7 p8	pcline
	p5	pcline	p6 p7 p8
	p6	r_lbinsen p5	p1
	p7	p5 r_lbinsen	p1
	p8	p5 r_lbinsen	pl
carga de rótulos	p1	p21	pcline
	p9	pcline	p21
	p21	xfillab p9	p1
actualização do estado dos sensores	p10	p18 xcbotbs	r_lbinsen
	pll	pl8 xlboths	r_lbinsen
	pl8	xlbpsupp	p10 pll pl9 p20 p26
	pl9	xlbfs pl8	r_lbinsen
	p20	xlbjs p18	r_lbinsen
	p26	xlbtbs pl8	r_lbinsen
informação para o estado da linha	p1	p6 p22	pcline
	p5	pcline	p6 p22

Tabela 4.4.1 - Modelo Funcional da UP do SEM(continuação)

Funcionalidade (*use-case*)	**Modelo das funções**		
	Processos	**Origem da informação**	**Destino da informação**
	p6	p5 r_lbinsen	p1
	p22	p5 r_lbinsen	p1
aviso de variação de velocidade	p23	p4	xlbwdev
aviso da situação da máquina	p24	p3 p22	p25
	p25	p6 p24	xlbdy

Tabela 4.4.1 - Modelo Funcional da UP do SEM(continuação)

N°	Função do processo no sistema	Nome do processo	Nº atrib	Nome do atributo
p1	comunicação entre a unidade de processamento c o pc da linha	lbcmpcl	atl	ccolbpcl
p2	descodificação do sinal de *stop* para a rotuladora	lbstd2	at2	clbstl2
p3	paragem da rotuladora	lbstp3	at3	csüb3
p4	controlo de velocidade da rotuladora	lbspcn-4	at4	clbctr4
p5	descodificação do endereço da rotuladora	lbdad5	at5	clbsel5
p6	leitura do estado do sensor de congestionamento da rotuladora	lbrjs6	at6	clbjst6
p7	leitura do estado do sensor de ritmo da rotuladora	lbrbr7	at7	clbbotrt7
p8	leitura do estado do sensor de nível da rotuladora	lbnbbs	at8	cnbollbb8
p9	descodificação do sinal "linhaok" para produção	lbdlok9	at9	clblok9
p10	acondicionamento do sinal do sensor de ritmo da rotuladora	lbcdbr10	at10	cibbrt10
Pll	acondicionamento do sinal do sensor de nível da rotuladora	lbcdbbll	atl l	ccsnbl11
p12	referencia de velocidade da rotuladora	lbspdr12	atl2	cbasplbl2
p13	aumento da velocidade da rotuladora	lbinspl3	atl3	dbinsprl3
pl4	diminuição da velocidade da rotuladora	lbdespl4	atl4	clbdesprl4
pl5	referência de velocidade para a marcha lenta à frente da rotuladora	lbslfwrl5	atl5	clbslfsl5
pl6	referência de velocidade para a marcha lenta atrás da rotuladora	lbsirer16	atl6	clbslrsl6
pl7	descodificação do sinal de linha em configuração para a rotuladora	lbdsup17	atl7	cllstupl7
pl8	actualização do estado dos sensores	lbtupsst18	atl8	dbupsst18
pl9	acondicionamento do sinal do sensor de encravamento da rotuladora	lbstfsl9	atl9	clbflstl9
p20	acondicionamento do sinal do sensor de congestionamento da rotuladora	lbstjs20	at20	clbjmst20
p21	carga de rótulos da rotuladora	lbfill21	at21	cfillab21
p22	leitura do estado do sensor de encravamento da rotuladora	lbrfs22	at22	clbfst22
p23	detecção de variação de velocidade da rotuladora	lbschg23	at23	dlbspch23
p24	detecção de avaria ou paragem da rotuladora	lbfst24	at24	clstpfl24
p25	acondicionamento do sinal da situação da rotuladora para o seu visualizador	lbcond25	at25	dlbst25
p26	acondicionamento do sinal do número total de garrafas processadas	lbtpbt26	at26	dtprcbt26

Tabela 4.4.2 - Dicionário dos processos e atributos do DFD da UP do SEM

nome do actor	função	atributo
xlbnst	botão de paragem normal (paragemnormal)	cibnstp
xlbsif	botão de marcha lenta à frente (botâomarchalentafrente)	cibsfw
xlbsir	botão de marcha lenta atrás (botãomarchalentaatrás)	cibsire
xfiliab	botão de carga de rótulos (botãocargarótulos)	cfilabcm
xlbpsupp	fonte de alimen. da unidade de process. (fontealimentuniproc)	cibpon
pcline	PC da linha (PCLinha)	copcpu
elbolb	quadro eléctrico da máquina (quadroeléctricomáquina)	cibemgst
rlbinsen	registo do estado dos sensores (registoestadosensores)	cregout
xlbfs	sensor de encravamento (sensorencravamento)	cflib
xlboths	sensor de ritmo (sensorritmo)	cibbhs
xcbotbs	sensor de nível (sensornível)	cibbotb
xlbjs	sensor de congestionamento (sensorcongestionamento)	cjmlb
xlbtbs	sensor de contagem (sensorcontagem)	dtprbt
xlbwdev	sinalizador da variação de velocidade (sinalizador)	------
Ibctil	unidade de controlo de velocidade (controlomotor)	------
xlbdy	visor da situação da máquina (visor)	------

Tabela 4.4.3 - Dicionário dos actores que interagem com a UP do SEM

Deste modo, com base nas tabelas 4.4.2 e 4.4.3, é possível fazer uma interpretação fácil do DFD do sistema descrito na tabela 4.4.1, a partir das funcionalidades dos seus *use-cases*.

Para efeitos de partição assumiu-se que:

1. as tarefas tinham complexidades e tempos de processamento semelhantes, e
2. as mensagens tinham o mesmo comprimento de palavra.

Por consequência, foram atribuídos pesos iguais a todas as tarefas do sistema e, respectivas interligações.

Desde já pode observar-se que a funcionalidade descrita no DFD generalizado (e que se pretende encapsular num número restrito de objectos coesos e autónomos) é complexa. Relativamente aos *use-cases*, nota-se que existem 7 processos, de entre os 26 processos atómicos identificados (p1, p4, p5, p6, p10 e p11), que são utilizados por mais de um *use-case*. O sucesso de processo de geração de uma arquitectura-alvo reside precisamente na capacidade de identificação desses objectos (provavelmente, em número inferior ao número de *use-cases*) com as características desejadas.

A partir do DFD construído anteriormente, o `SysObj` gera o grafo GPA, que mapeia aquele DFD, e com base nele cria o grafo GOT. A partir do GOT são identificadas todas as partições possíveis para o sistema (unidade de processamento do SEM da máquina rotuladora). Para tal, utiliza-se a opção *curve-tracer* do `SysObj`. Posteriormente, com base nas métricas e critérios de decisão propostos nesta tese, procede-se à selecção da arquitectura-alvo para o presente sistema.

Selecção Grosseira :

Nas figuras 4.33 e 4.34 representa-se o comportamento das métricas de autonomia média total, número total de malhas, coesão média total e acoplamento total do sistema, para o universo de arquitecturas que podem implementar a UP do SEM da máquina rotuladora.

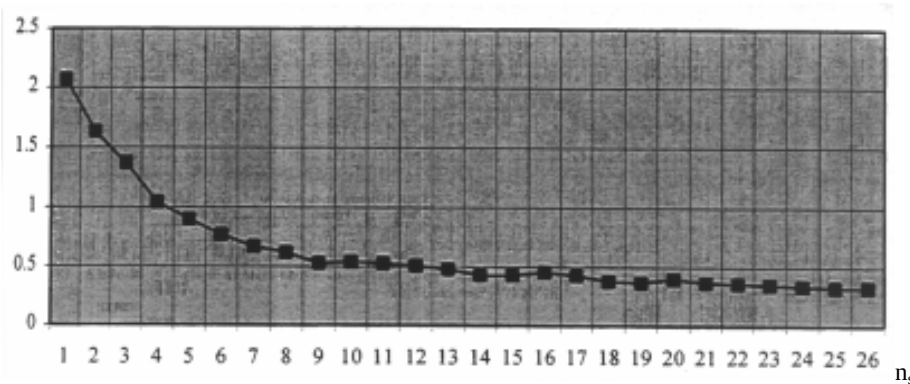

Figura 4.4.33 - Autonomia média total (*ATAt*) para o conjunto de arquitecturas possível

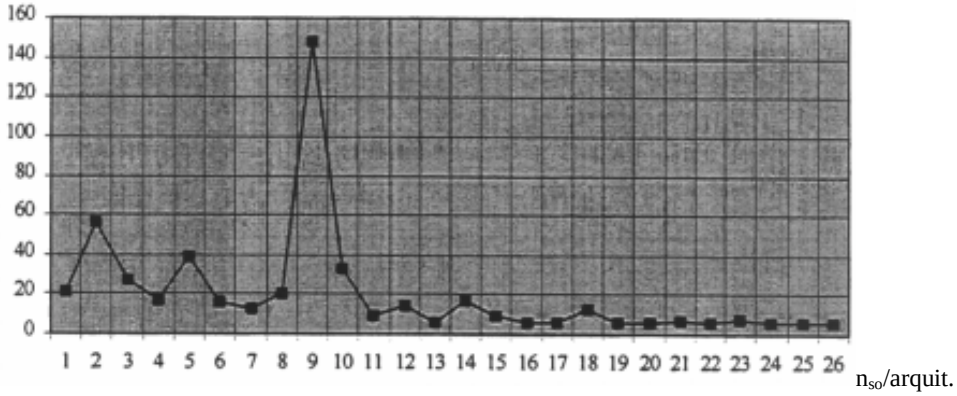

Figura 4.4.34 - Número de malhas (nl_t) para o conjunto de arquitecturas possível

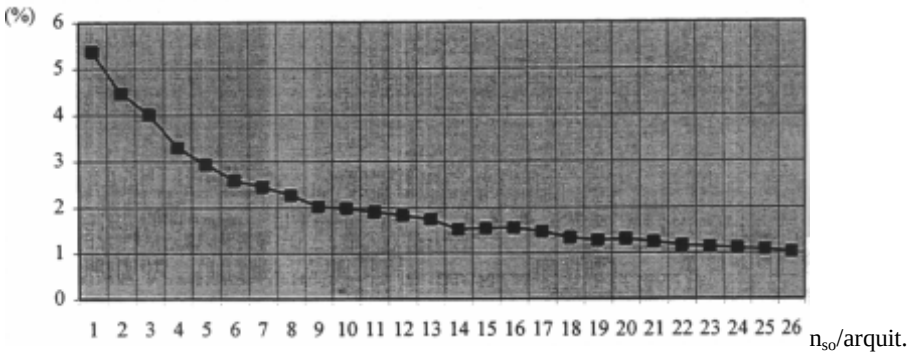

Figura 4.4.35 - Coesão média total (*ATCoh*) para o conjunto de arquitecturas possível

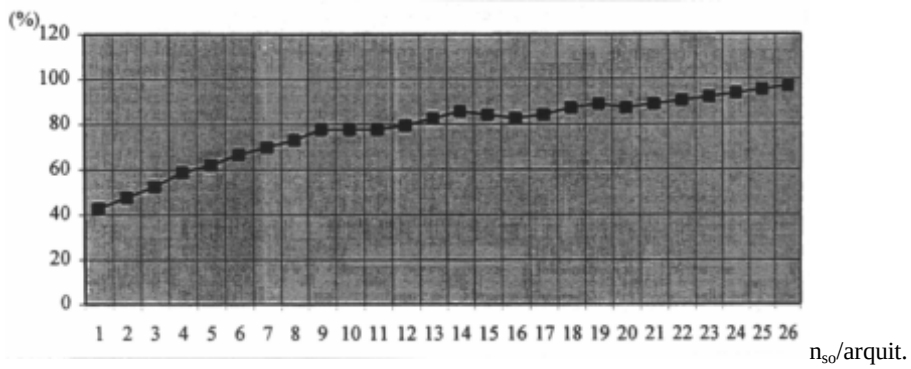

Figura 4.4.36 - Acoplamento total do sistema (*ASTCp*) para o conjunto de arquitecturas possível

Verifica-se que as arquitecturas com quatro e cinco objectos são as que reúnem melhores condições para a implementação do sistema. De facto, como se resume na tabela 4.4.4, os valores das métricas utilizadas para a discriminação grosseira da selecção das arquitecturas apresentam valores muito semelhantes. A diferença do número de malhas para a arquitectura de quatro objectos, não é razão suficiente para ser ela a seleccionada. Devem considerar-se também os valores das métricas de testabilidade intrínseca do sistema, e o equilíbrio da distribuição das tarefas pelos objectos, o que será feito na Selecção Fina.

Métricas	**Arquitecturas**	
	4 Objectos	**5 Objectos**
autonomia média total (atat)	1.04%	0.9%
número total de malhas (nlt)	17%	39%
coesão média total (atcoh)	3.3%	2.94%
acoplamento total do sistema (stcp)	58.73%	61.9%

Tabela 4.4.4 – Valores das métricas de desempenho

Selecção Fina :

Com o objectivo de realizar a discriminação fina para a selecção da arquitectura-alvo que implementará o controlo (UP) da máquina rotuladora, mostram-se na tabela 4.5 relativamente ao sistema, os valores das métricas de controlabilidade e observabilidade médias, e relativamente aos objectos, os valores do desvio padrão das métricas de testabilidade, e das métricas de coesão, autonomia, peso relativo e acoplamento direccional (*fan out*).

Métricas e Desvios Padrão	**Arquitecturas**	
	4 Objectos	**5 Objectos**
Controlabilidade média de fronteira do sistema (*ASBC*)	55.41%	56.79%
Observabilidade média de fronteira (*ASBO*)	37.50%	26.67%
Controlabilidade média do sistema (*ASCtr*)	34.86%	43.30%
Observabilidade média do sistema (*ASObs*)	16.07%	14.67%
Desvio padrão de controlabilidade de fronteira dos Objectos (*ST-OBC*)	27.52%	38.74%
Desvio padrão da observabilidade de fronteira dos objectos (*ST-OBO*)	37.50%	24.94%
Desvio padrão da controlabilidade dos objectos (*ST-OCtr*)	17.99%	32.97%
Desvio padrão da observabilidade dos objectos (*ST-Oobs*)	11.83%	12.93%
Desvio padrão da coesão total dos objectos (*ST-Tcoh*)	0.71%	0.39%
Desvio padrão da autonomia total dos objectos (*ST-CAt*)	0.20%	0.21%
Desvio padrão do peso relativo dos objectos (*ST-ORW*)	1.92%	1.54%
Desvio padrão do acoplamento direccional dos objectos (*ST-DTCp*)	5.64%	4.49%

Tabela 4.4.5 – Valores das métricas de testabilidade intrínseca e desvios padrão

Da análise da tabela 4.4.5 pode concluir-se que não existem valores que conduzam a uma opção óbvia quanto à selecção da arquitectura. De facto, se a controlabilidade média de fronteira e a controlabilidade média do sistema apresentam valores favoráveis à arquitectura de 5 objectos, com as diferenças percentuais de 1.38 e 8.84, respectivamente, já os valores da observabilidade de fronteira e da observabilidade média do sistema são favoráveis à arquitectura de 4 objectos, apresentando a vantagem de 10.83 e 1.4 pontos percentuais, respectivamente. Quanto ao desvio padrão correspondente, que é uma medida do equilíbrio dos factores de mérito das

arquitecturas, verifica-se que os valores da observabilidade de fronteira dos objectos, da coesão total dos objectos, do peso relativo dos objectos e do acoplamento direccional dos objectos apontam para a arquitectura de 5 objectos, com diferenças que variam entre 12.56 e 0.38. Porém, os valores dos desvios padrão da controlabilidade de fronteira dos objectos, da controlabilidade do objecto, da observabilidade do objecto e da autonomia total dos objectos, são favoráveis à selecção da arquitectura de 4 objectos, com valores que variam entre 14.98 e l pontos percentuais. Deste modo, o facto da arquitectura de 4 objectos ter 17 malhas e a de 5 objectos apresentar 39 malhas, toma-se o critério determinante apontando para a selecção da arquitectura de quatro objectos. Por conseguinte esta é a arquitectura seleccionada (arquitectura-alvo) para a implementação da UP do SEM da máquina rotuladora do sistema *Fill*.

Objectos :

Na tabela 4.4.6 apresenta-se a lista dos atributos da arquitectura, com os respectivos agentes origem e destino. Na tabela 4.4.7 listam-se os objectos e actores que colaboram para a implementação de cada um dos *use-cases* do sistema.

Como já foi referido, dadas as restrições físicas impostas à realização do sistema, os objectos da arquitectura serão realizados **em** *software*, e atribuídos ao nó constituído por uma plataforma de *hard-ware* (microprocessador) com as características necessárias para comportar o código correspondente à implementação da funcionalidade dos objectos e com as interfaces adequadas para os sinais dos sensores, controlo da máquina e comunicação com o computador de linha.

Agente origem	Atributo	Agente destino
pcline	copcpu	objl
pcline	copcpu	obj3 (processo p5)
pcline	copcpu	obj3 (processo p9)
pcline	copcpu	obj4
elbpolb	ibemgst	obj4
xcbotbs	ibbotb	obj2
xfiliab	filabcm	obj3
xlbfs	fllb	obj2
xlbfs	fllb	obj4
xlbjs	jmlb	obj4
xlbnst	ibnstp	obj4
xlboths	ibbhs	obj2
xlbpsupp	ibpon	obj2
xlbsif	ibsifw	objl
xlbsir	ibsire	objl
xlbtbs	tprbt	obj2
obj3	colbpcl	pcline
obj4	lbst25	xlbdy
objl	lbspch23	xlbwdev
objl	lbctr4	ibctil
obj2	csnblll	objl
obj2	lbinsprl3	objl
obj2	basplbl2	objl
obj2	lbupsstl8	obj4
obj3	lbsel5	obj4
obj3	lblok9	objl
obj3	lbfst22	obj4
obj4	lbjst6	obj3
obj4	stlb3	objl
obj4	stlb3	obj3
obj2	csnblll	r_ibinsen
obj2	lbbrt10	r_lbinsen
r_lbinsen	regout	obj3 (processo p7)
r_lbinsen	regout	obj3 (processo p8)

Tabela 4.4.6 - Lista dos agentes de origem e destino do SEM

Agente origem	Atributo	Agente destino
r_ibinsen	regout	obj3 (processo p22)
r_lbinsen	regout	obj4
obj2	lbflstl9	r_lbinsen
obj4	lbjmst20	r_lbinsen
obj2	tprcbt26	r_lbinsen

Tabela 4.4.6 - Lista dos agentes de origem e destino do SEM (continuação)

Use-cases	**Objectos**	**Actores**
paragem da máquina	objl obj3 obj4	pcline elbolb xlbfs xlbnst
máquina em produção	objl obj2 obj3	pcline xlboths xcbotbs ibctll
configuração da máquina	objl	pcline xlbsif xlbsir ibstll
correcção da velocidade *versus* congestionamento na saída	obj3 obj4	pcline r_lbinsen
carga de rótulos	obj3	pcline xfiliab
actualiza estado dos sensores	obj2 obj4	xlbfs xlbtbs xlbpsupp xlbjs r_lbinsen xcbotbs xlboths
informação para o estado da linha	obj3 obj4	pcline r_lbinsen
aviso de variação de velocidade	objl	xlbwdev
aviso da situação da máquina	obj4	xlbdy

Tabela 4.4.7 - Lista de colaboração para a realização dos *use-cases*

Objectos	Métricas					
	coesão *(TCohk)*	autonomia *(TAtk)*	acoplamento *(STCp)*	acopl. direcc. *(DTCpk)*	contr. front *(OBCt)*	obs. front. *(OBOk)*
objl	2.82 %	1.10	58.73%	10.53 %	37.50 %	100.00%
obj2	4.14%	1.08		15.79%	100.00 %	0.00%
obj3	3.85%	1.25		15.79 %	28.57 %	25.00%
obj4	2.40 %	0,71		21.05%	55.56 %	25.00%

Tabela 4.4.8 — Características da arquitectura-alvo

Como se pode observar, a solução a que foi conduzido assegura uma arquitectura com quatro objectos autónomos, com uma distribuição equilibrada, acoplamento direccional, e com a coesão e acoplamento de sistemas moderados.

A partilha da funcionalidade por diferentes *use-cases é* novamente evidente na tabela 4.7, já que todos os objectos participam da funcionalidade de mais de um *use-case.* A título exemplificativo, o objl é utilizado por 4 *use-cases*, três deles através do processo p4, e um através do processo p23.

Igualmente se pode verificar (da tabela 4.8) que as condições suficientes de testabilidade não estão asseguradas, já que os diferentes objectos não são directamente controláveis e observáveis (objl é directamente observável, e obj2 é directamente controlável) desde o exterior do sistema. Assim, se este for um requisito do sistema, ter-se-ia de aplicar uma técnica DFT e fazer a reconfiguração da arquitectura, com análise de compromisso.

4.5. Conclusões

4.5.1. Trabalho Realizado

Foi desenvolvida e incorporada no `SysObj` uma ferramenta de desenho gráfico para diagramas de fluxo de dados. A introdução da facilidade de desenho no `SysObj`, permite fazer com que seja uma ferramenta independente de outras aplicações, como o `ParadimPlus.` Porém a importação dos ficheiros *.cdf do `ParadigmPlus` permanece activa. No desenvolvimento da interface gráfica foi mantida a estrutura dos projectos do `SysObj`. Deste modo, um projecto continua a poder conter vários módulos sendo os diagramas de fluxo de dados desenhados separadamente para cada módulo.

Foi também incluída uma funcionalidade de geração dos *use-cases* implementados. Esta funcionalidade permite ver os diagramas desenhados sob a forma de *use-case*, com as área de implementação a representarem processos e as setas orientadas de um use-case para o outro a serem os atributos que entram e saem de cada área de implementação ou seja de cada funcionalidade do *use-case*. Por motivos técnicos não foi implementado o desenho dos actores, externos e internos, na interacção com os *use-cases*. Actores esses que completam a correcta representação de um modelo use-case.

Foi ainda desenvolvido este manual que descreve tecnicamente a interface desenvolvida e ainda apresenta um exemplo completo de utilização do `SysObj`, um manual do utilizador onde são focados pontos como as ferramentas disponíveis, a forma de utilização, o significado dos objectos envolvidos, a interpretação de resultados, etc.

É ainda incluído um manual de instalação do `SysObj 4.0`, o qual esclarece algumas dúvidas que possam surgir durante a instalação.

O projecto é apresentado na forma de CD contendo todos os manuais (incluindo os manuais da primeira versão), o código fonte do programa e as versões de instalação do mesmo.

4.5.2. Perspectivas de Trabalho Futuro

A interface gráfica desenvolvida veio proporcionar uma maior independência e comodidade, em termos de utilização de ferramentas, ao projectista, que pode agora desenhar os diagramas de fluxo de dados no ambiente de trabalho do `SysObj`. Porém, a ferramenta pode ainda ser melhorada, nomeadamente no que concerne :

à exportação dos diagramas de fluxo de dados para os ficheiros *.cdf :
O facto de esta interface tornar o `SysObj` independente de programas como o `Paradigm Plus`, não é factor suficiente para a tornar fechada a outras ferramentas, isto é, a ligação com outras ferramentas não deve ser quebrada. Assim a exportação dos diagramas desenvolvidos para ficheiros *.cdf torna-se uma funcionalidade necessária;

ao desenho de diagramas de fluxo de dados a partir de um ficheiro *.cdf :
Pela razão apresentada anteriormente, o `SysObj` deve continuar a aceitar os ficheiros *.cdf para importação. Assim torna-se interessante que o `SysObj` represente gráficamente os diagramas importados dos ficheiros *.cdf. Esta funcionalidade teria uma elevada utilidade, mas também um elevado grau de dificuldade na sua implementação, uma vez que o programa teria que ser inteligente ao dispor os objectos no ecrã, uma vez que os ficheiros *.cdf apenas trazem informação relativa aos objectos em si e não relativamente á sua localização e tamanho.

à possibilidade de ver os *use-cases* completos dos diagramas desenhados :

Os *use-case*s dos diagramas de fluxo de dados que estão a ser representados apenas mostram os processos e os atributos de entrada e saída de cada um. Assim, é de interesse visualizar também os actores, externos e internos, representados para dar uma ideia completa de como eles contribuem para o *use-case* que se está a representar.

à visualização gráfica do modelo de objectos gerado:
É de interesse que o projectista, no fim de realizar as partições, possa analisar os resultados numa forma gráfica, isto é, com os objectos gerados desenhados e interligados. Em termos técnicos esta funcionalidade também tem um grau de dificuldade elevado, porém poderia ser utilizado o mesmo "motor" de inteligência desenvolvido para desenho de diagramas de fluxo de dados a partir de ficheiros *.cdf.

Compatibilização com sistemas EDA comerciais :
Um dos factores críticos de sucesso de uma metodologia e das ferramentas que a suportam é a possibilidade de transferência, sem discontinuidade, de informação para sistemas CASE e EDA já existentes, e com os quais as equipas de projecto de *software* e *hardware* estão familiarizadas. Assim, o estabelecimento de interfaces com sistemas como o `Rational Rose`, o `Objectime` e o sistema de síntese de *hardware* `Synopsys` deveria ser ponderado.

Bibliografia

[EP98] Hans-Erik Erikson and Magnus Penker, "UML Toolkit", Jonh Wiley & Sons, Inc., 1998

[FS97] Martin Fowler with Kendal Scott, "UML Distilled – Applying the Standard Object Modeling Language", Addison-Wesley Longman, Inc., 1997

[R97a] Rui M. V. Ribeiro, "Sistema Electrónico para Automatização de Linha de Produção Vinícola", Tese de Mestrado em Engenharia Electrónica e de Computadores, IST, Outubro, 1997.

[RDTT98] R Ribeiro, O. P. Dias, J. P. Teixeira and I.C. Teixeira, "Automation and Real-Time Control of a Wine Bottling Production Line – And Object-Oriented Based Hardware/Software Co-Design Case Study", Preprints of the 8th IFAC/IFORS/IMACS/IFIP Symposium on Large Scale Systems : Theory & Applications, vol. II, pp. 1211-1216, July, 1998.

[RJB99] James Rumbaug, Ivar Jacobson and Grady Booch, "The Unified Modeling Language Reference Manual", Addyson- Wesley Longman, Inc., 1999.